Oracle® Retail Merchandising Foundation Cloud Service/Merchandising System

Operations Guide, Volume 1 - Batch Overviews and Designs

Release 19.0

F24698-03

June 2020

ORACLE®

Oracle® Retail Merchandising Foundation Cloud Service/Merchandising System Operations Guide, Volume 1 - Batch Overviews and Designs, Release 19.0

F24698-03

Primary Author: Alex Meske

enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

Send Us Your Comments lv

Preface lvii

Audience lvii
Documentation Accessibility lvii
Related Documents lvii
Customer Support lviii
Improved Process for Oracle Retail Documentation Corrections lviii
Oracle Retail Documentation on the Oracle Technology Network lix
Conventions lix

1 Introduction

Contents of This Guide 1-1
Volume 1 - Batch Overviews and Designs 1-1
Volume 2 - Message Publication and Subscription Designs 1-1
A Note about 'External' Subscription RIB APIs 1-1
Volume 3 - Back-End Configuration and Operations 1-2
Merchandising Modules 1-2
Batch Schedule 1-2
Pro *C Input and Output Formats 1-2
General Interface Discussion 1-2
Standard File Layouts 1-3
Detail-Only Files 1-3
Master and Detail Files 1-3

2 Administration Batch

Program Summary 2-1
async_job_status_retry_cleanup.ksh (Purge Asynchronous Job Tables) 2-1
Schedule 2-2
Design Overview 2-2
Restart/Recovery 2-2
Key Tables Affected 2-2
Input/Out Specification 2-2
prepost (Pre/Post Helper Processes for Batch Programs) 2-2
Schedule 2-3

Design Overview 2-3
Restart/Recovery 2-5
dlyprg (Daily Purge of Foundation Data) 2-5
Schedule 2-5
Design Overview 2-5
Restart Recovery 2-6
I/O Specification 2-6
Design Assumptions 2-6
daily_purge_job (Daily Purge of Foundation Data) 2-6
Design Overview 2-6
Scheduling Constraints 2-7
Restart Recovery 2-7
Key Tables Affected 2-7
I/O Specification 2-11
taxevntprg (Tax Event Purge) 2-11
Schedule 2-11
Design Overview 2-11
Restart/Recovery 2-11
Key Tables Affected 2-12
Design Assumptions 2-12
tax_event_purge_job (Tax Event Purge) 2-12
Design Overview 2-12
Scheduling Constraints 2-12
Restart/Recovery 2-13
Key Tables Affected 2-13
Input/Output Specification 2-13
dtesys (Increment Virtual Business Date) 2-13
Schedule 2-13
Design Overview 2-13
Restart/Recovery 2-14
I/O Specification 2-14
Design Assumptions 2-14
trunctbl.ksh (Truncate Table Script) 2-14
Schedule 2-14
Design Overview 2-15
Restart/Recovery 2-15
Key Tables Affected 2-15
Design Assumptions 2-15
rms_oi_purge.ksh (Purge Dashboard Working Tables) 2-15
Schedule 2-15
Design Overview 2-15
Restart/Recovery 2-16
Key Tables Affected 2-16
Design Assumptions 2-16
raf_notification_purge.ksh (Purge RAF Notifications) 2-17
Schedule 2-17
Design Overview 2-17

Restart/Recovery 2-17
Design Assumptions 2-17
batch_archive_purge_hist.ksh (Archive and Truncate Purge History Tables) 2-17
Schedule 2-17
Design Overview 2-18
Restart/Recovery 2-18
I/O Specifications 2-18
Design Assumptions 2-18
admin_api_purge.ksh (Purge Manage Admin Records) 2-18
Schedule 2-19
Design Overview 2-19
Restart/Recovery 2-19
I/O Specifications 2-19
refreshmview.ksh (Forecast Roll Up Refresh Views) 2-19
Design Overview 2-19
Scheduling Constraints 2-19
Restart/Recovery 2-20
Key Tables Affected 2-20
I/O Specification 2-20
data_export_purge_job (Purging of All the Extracted Data) 2-20
Design Overview 2-20
Scheduling Constraints 2-21
Restart/Recovery 2-21
Key Tables Affected 2-21
Integration Contract 2-21
Design Assumptions 2-21
job_audit_logs_purge_job (Purge Old Job Auditing Logs) 2-21
Design Overview 2-22
Scheduling Constraints 2-22
Restart/Recovery 2-22
Key Tables Affected 2-22
Design Assumptions 2-23

3 Foundation Data Maintenance

Batch Design Summary 3-2
admin_api_purge (Purge Manage Admin records) 3-2
Schedule 3-3
Design Overview 3-3
Restart/Recovery 3-3
I/O Specification 3-3
batch_compeffupd (Update ELC Components) 3-3
Schedule 3-3
Design Overview 3-3
Restart/Recovery 3-4
Design Assumptions 3-4
batch_expprofupd (Apply Pending Rate Changes to Expense Profiles) 3-4
Schedule 3-4

Design Overview ... 3-5
Restart/Recovery ... 3-5
Design Assumptions ... 3-5
batch_depchrgupd (Apply Pending Up-Charge Cost Component Changes to Departments) ... 3-5
Schedule ... 3-5
Design Overview ... 3-5
Restart/Recovery ... 3-5
Design Assumptions ... 3-5
batch_itmcostcompupd (Apply Pending Item Cost Component Updates) ... 3-6
Schedule ... 3-6
Design Overview ... 3-6
Restart/Recovery ... 3-6
Design Assumptions ... 3-6
batch_alloctsfupd (Update Allocation and Transfer Based on Changes to Up-Charges) ... 3-6
Schedule ... 3-7
Design Overview ... 3-7
Restart/Recovery ... 3-7
Design Assumptions ... 3-7
batch_ordcostcompupd (Apply Pending Cost Component and ELC Changes to Purchase Orders) 3-7
Schedule ... 3-7
Design Overview ... 3-7
Restart/Recovery ... 3-8
Design Assumptions ... 3-8
elcexcprg (Purge Aged Cost Component Exceptions) ... 3-8
Schedule ... 3-8
Design Overview ... 3-8
Restart/Recovery ... 3-9
Design Assumptions ... 3-9
elc_except_purge_job (Purge Aged Cost Component Exceptions) ... 3-9
Design Overview ... 3-9
Scheduling Constraints ... 3-10
Restart/Recovery ... 3-10
Key Tables Affected ... 3-10
Design Assumptions ... 3-10
dfrtbld (Build Diff Ratios Based on Sales History) ... 3-10
Schedule ... 3-11
Design Overview ... 3-11
Restart/Recovery ... 3-11
Key Tables Affected ... 3-11
I/O Specification ... 3-11
Output File Layout ... 3-11
Design Assumptions ... 3-12
lclrbld (Rebuild Dynamic Location Lists) ... 3-12
Schedule ... 3-12
Design Overview ... 3-12
Restart/Recovery ... 3-12
Key Tables Affected ... 3-12

Design Assumptions ... 3-13
loc_list_rebuild_job (Rebuild Dynamic Location Lists) ... 3-13
Design Overview ... 3-13
Scheduling Constraints ... 3-13
Restart/Recovery ... 3-14
Key Tables Affected ... 3-14
Design Assumptions ... 3-14
batch_rfmvcurrconv (Refresh Currency Conversion Materialized View) ... 3-14
Schedule ... 3-14
Design Overview ... 3-14
Restart/Recovery ... 3-14
Key Tables Affected ... 3-15
Design Assumptions ... 3-15
refmvlocprimaddr (Refresh Address Materialized View) ... 3-15
Schedule ... 3-15
Design Overview ... 3-15
Restart/Recovery ... 3-15
Key Tables Affected ... 3-15
Design Assumptions ... 3-16
cremhierdly (Process Pending Merchandise Hierarchy Changes from External Systems) ... 3-16
Schedule ... 3-16
Design Overview ... 3-16
Restart/Recovery ... 3-16
Key Tables Affected ... 3-16
Design Assumptions ... 3-17
reclsdly (Reclassify Items in Merchandise Hierarchy ... 3-17
Schedule ... 3-17
Design Overview ... 3-17
Restart/Recovery ... 3-17
Key Tables Affected ... 3-17
Design Assumptions ... 3-18
supmth (Rollup of Supplier Data) ... 3-18
Schedule ... 3-19
Design Overview ... 3-19
Restart/Recovery ... 3-19
Key Tables Affected ... 3-19
Design Assumptions ... 3-19
schedprg (Purge Aged Store Ship Schedule) ... 3-19
Schedule ... 3-20
Design Overview ... 3-20
Restart/Recovery ... 3-20
Key Tables Affected ... 3-20
Design Assumptions ... 3-20
activity_sched_purge_job (Purge Aged Store Ship Schedule) ... 3-20
Design Overview ... 3-21
Scheduling Constraints ... 3-21
Restart/Recovery ... 3-21

Key Tables Affected 3-21
Design Assumptions 3-22
prchstprg(Purge Aged Price History Data) 3-22
Schedule 3-22
Design Overview 3-22
Restart/Recovery 3-22
Performance Considerations 3-23
Key Tables Affected 3-23
Design Assumptions 3-23
price_hist_purge_job (Purge Aged Price History Data) 3-23
Design Overview 3-23
Scheduling Constraints 3-24
Restart/Recovery 3-24
Key Tables Affected 3-24
tcktdnld (Download of Data to be Printed on Tickets) 3-24
Schedule 3-25
Design Overview 3-25
Restart/Recovery 3-25
Key Tables Affected 3-25
I/O Specification 3-26
Output File Layout 3-26
Design Assumptions 3-28
refmvl10entity (Refresh MV MV_L10N_ENTITY) 3-28
Schedule 3-29
Design Overview 3-29
Restart/Recovery 3-29
Locking Strategy 3-29
Security Considerations 3-29
Performance Considerations 3-29
I/O Specification 3-29
likestorebatch (Like Store Batch Processing) 3-29
Schedule 3-30
Design Overview 3-30
Restart/Recovery 3-30
Key Tables Affected 3-30
Design Assumptions 3-31
straddbatch.ksh(Store Add Asynchronous Process) 3-31
Business Overview 3-31
Key Tables Affected 3-31
Design Assumptions 3-32
Queue Creation 3-32
Design Overview - Process Steps 3-32
Running entire store-add as batch in case of AQ issues 3-33
Building Schedule Dependencies between Async process and other batches 3-33
Monitoring Progress of Store-Add Processes 3-33
CORESVC_STORE_ADD_SQL. ADD_STORE (Store Add Asynchronous Process) 3-33
Business Overview 3-33

Key Tables Affected 3-34
Design Assumptions 3-34
Queue Creation 3-34
Design Overview - Process Steps 3-35
Package Impact 3-35
Function Level Description - ADD_STORE 3-35
Function Level Description - ENQUEUE_STORE_ADD 3-36
Function Level Description - ENQUEUE_STORE_ADD_RETRY 3-36
Function Level Description - NOTIFY_STORE_ADD 3-36
Operations and Monitoring 3-36
Running entire Store-Add as Batch in Case of AQ Issues 3-37
Building Schedule Dependencies between Async Process and other Batches 3-37
Monitoring Progress of Store-Add Processes 3-37

4 Item Maintenance

Program Summary 4-1
sitmain (Scheduled Item Maintenance) 4-1
Schedule 4-1
Design Overview 4-1
Restart/Recovery 4-2
Key Tables Affected 4-2
Design Assumptions 4-2
vatdlxpl (Mass VAT Updates for Items/Locations) 4-2
Schedule 4-2
Design Overview 4-2
Restart/Recovery 4-3
Key Tables Affected 4-3
Design Assumptions 4-3
itm_indctn_purge.ksh (Purge Item Induction Staging Tables) 4-3
Design Overview 4-3
Scheduling Constraints 4-4
Restart/Recovery 4-4
Key Tables Affected 4-4
item_loc_purge_job (Daily Purge of Item-Location Data) 4-5
Design Overview 4-6
Scheduling Constraints 4-6
Restart/Recovery 4-6
Key Tables Affected 4-6
I/O Specification 4-7

5 Purchase Order

Batch Design Summary 5-1
edidlord (Download of Purchase Orders from Merchandising to Suppliers) 5-1
Schedule 5-2
Design Overview 5-2
Restart/Recovery 5-2

I/O Specification 5-2
Output File Layout 5-3
Design Assumptions 5-11
ediupack (Upload Purchase Order and Purchase Order Change Acknowledgements from Suppliers to Merchandising) 5-11
Schedule 5-11
Design Overview 5-11
Restart/Recovery 5-12
I/O Specification 5-12
Input File Layout 5-12
Design Assumptions 5-14
vrplbld (Build Purchase Orders for Vendor Generated Orders) 5-14
Schedule 5-14
Design Overview 5-15
Restart/Recovery 5-15
Key Tables Affected 5-15
Design Assumptions 5-16
genpreiss (Generate Pre-Issued Order Numbers) 5-16
Schedule 5-16
Design Overview 5-16
Restart/Recovery 5-16
Design Assumptions 5-16
supcnstr (Scale Purchase Orders Based on Supplier Constraints) 5-17
Schedule 5-17
Design Overview 5-17
Restart/Recovery 5-17
Locking Strategy 5-17
Key Tables Affected 5-18
Design Assumptions 5-18
orddscnt (Apply Deal Discounts to Purchase Orders) 5-18
Schedule 5-18
Design Overview 5-18
Restart/Recovery 5-18
Design Assumptions 5-18
ordupd (Update Retail Values on Open Purchase Orders) 5-19
Schedule 5-19
Design Overview 5-19
Restart/Recovery 5-19
Design Assumptions 5-19
ordautcl (Auto Close Purchase Orders) 5-19
Schedule 5-19
Design Overview 5-20
Category 1 5-20
Category 2 5-20
Category 3 5-20
Restart/Recovery 5-20
Design Assumptions 5-21
order_auto_close_job (Auto Close Purchase Orders) 5-21

Design Overview 5-21
Category 1 5-21
Category 2 5-21
Category 3 5-21
Scheduling Constraints 5-22
Restart/Recovery 5-22
Key Tables Affected 5-22
Design Assumptions 5-24
ordrev (Write Purchase Order Information to Purchase Order History Tables) 5-24
Schedule 5-24
Design Overview 5-24
Restart/Recovery 5-25
Design Assumptions 5-25
order_revision_job (Write Purchase Order Information to Purchase Order History Tables) 5-25
Design Overview 5-25
Scheduling Constraints 5-26
Restart/Recovery 5-26
Key Tables Affected 5-26
Design Assumptions 5-26
ordprg (Purge Aged Purchase Orders) 5-27
Schedule 5-27
Design Overview 5-27
Restart/Recovery 5-27
Design Assumptions 5-28
order_purge_job (Purge Aged Purchase Orders) 5-28
Design Overview 5-28
Scheduling Constraints 5-29
Restart/Recovery 5-29
Key Tables Affected 5-29
Design Assumptions 5-33
poindbatch.ksh (Upload Order Data) 5-33
Schedule 5-33
Design Overview 5-33
Restart/Recovery 5-34
Design Assumptions 5-34
po_indctn_purge.ksh (Purge PO Induction Staging Tables) 5-34
Schedule 5-34
Design Overview 5-34
Restart/Recovery 5-35
Design Assumptions 5-35

6 Deals

Program Summary 6-1
dealupld (Upload of Deals from 3rd Party Systems) 6-2
Design Overview 6-2
Restart/Recovery 6-2
I/O Specification 6-2

Integration File Layout 6-2
Design Assumptions 6-28
batch_ditinsrt.ksh (Deal Calculation Queue Insert Multithreading) 6-28
Schedule 6-28
Design Overview 6-28
Restart/Recovery 6-28
Design Assumptions 6-28
ditinsrt (Insert into Deal Calculation Queue) 6-28
Schedule 6-29
Design Overview 6-29
Restart/Recovery 6-29
Design Assumptions 6-29
discotbapply (Update OTB After Deal Discounts) 6-29
Schedule 6-29
Design Overview 6-30
Restart/Recovery 6-30
Schedule 6-30
dealact (Calculate Actual Impact of Billback Deals) 6-30
Schedule 6-30
Design Overview 6-30
Restart/Recovery 6-30
Design Assumptions 6-30
dealinc (Calculate Weekly/Monthly Income Based on Turnover) 6-30
Schedule 6-31
Design Overview 6-31
Restart/Recovery 6-31
Design Assumptions 6-31
dealday (Daily Posting of Deal Income to Stock & General Ledgers) 6-31
Schedule 6-31
Design Overview 6-32
Restart/Recovery 6-32
Design Assumptions 6-32
dealfct (Calculates/Update Forecasted Values for Deals) 6-32
Schedule 6-32
Design Overview 6-32
Restart/Recovery 6-32
Design Assumptions 6-33
vendinvc (Stage Complex Deal Invoice Information) 6-33
Schedule 6-33
Design Overview 6-33
Restart/Recovery 6-33
I/O Specification 6-33
Design Assumptions 6-33
vendinvf (Stage Fixed Deal Invoice Information 6-34
Schedule 6-34
Design Overview 6-34
Restart/Recovery 6-34

I/O Specification 6-34
Design Assumptions 6-34
dealcls (Close Expired Deals) 6-34
Schedule 6-35
Design Overview 6-35
Restart/Recovery 6-35
Design Assumptions 6-35
deal_close_job (Close Expired Deals) 6-35
Design Overview 6-35
Scheduling Constraints 6-36
Restart/Recovery 6-36
Key Tables Affected 6-36
dealprg (Purge Closed Deals) 6-36
Schedule 6-37
Design Overview 6-37
Restart/Recovery 6-37
Design Assumptions 6-37
deal_purge_job (Purge Closed Deals) 6-37
Design Overview 6-37
Scheduling Constraints 6-38
Restart/Recovery 6-38
Key Tables Affected 6-38
deal_actuals_purge_job (Purge Closed Deals Actuals Item/Location) 6-39
Design Overview 6-39
Scheduling Constraints 6-40
Restart/Recovery 6-40
Key Tables Affected 6-40

7 Contracts

Batch Design Summary 7-1
edidlcon (Download Contracts to Suppliers) 7-2
Schedule 7-2
Design Overview 7-2
Restart/Recovery 7-2
I/O Specification 7-2
Output File Layout 7-3
Design Assumptions 7-7
ediupavl (Upload Item Availability for Type A & D Contracts from Suppliers) 7-7
Schedule 7-7
Design Overview 7-7
Restart/Recovery 7-8
I/O Specification 7-8
Input File Layout 7-8
Design Assumptions 7-9
cntrordb (Create Replenishment Orders for Item/Locations on Type B Contracts) 7-9
Schedule 7-10
Design Overview 7-10

Restart/Recovery 7-10
Design Assumptions 7-10
cntrprss (Apply Type A, C and D Contracts to Orders Created by Replenishment) 7-10
Schedule 7-11
Design Overview 7-11
Restart/Recovery 7-11
Design Assumptions 7-11
cntrmain (Contract Maintenance and Purging) 7-11
Schedule 7-11
Design Overview 7-11
Restart/Recovery 7-11
Design Assumptions 7-12
contract_purge_job (Contract Maintenance and Purging) 7-12
Design Overview 7-12
Scheduling Constraints 7-12
Restart/Recovery 7-13
Key Tables Affected 7-13
Design Assumptions 7-13

8 Cost Changes

Batch Design Summary 8-1
sccext (Supplier Cost Change Extract) 8-1
Schedule 8-1
Design Overview 8-1
Restart/Recovery 8-2
Design Assumptions 8-2
ccprg (Cost Change Purge) 8-2
Schedule 8-2
Design Overview 8-2
Restart/Recovery 8-2
Design Assumptions 8-2
cost_change_purge_job (Cost Change Purge) 8-3
Design Overview 8-3
Scheduling Constraints 8-3
Restart/Recovery 8-3
Key Tables Affected 8-4
Design Assumptions 8-4
ownership_change_process (Process Scheduled Ownership Change Data) 8-4
Schedule 8-4
Design Overview 8-4
Restart/Recovery 8-5
Restart/Recovery 8-5
Design Assumptions 8-6
ownership_change_purge (Purge Processed and Aged Ownership Change Data) 8-6
Schedule 8-7
Design Overview 8-7
Restart/Recovery 8-7

Key Tables Affected 8-7
Design Assumptions 8-7

9 Open To Buy

Batch Design Summary 9-1
otbdnld (Download Current & Future OTB by Subclass) 9-1
Schedule 9-2
Design Overview 9-2
Restart/Recovery 9-2
I/O Specification 9-2
Output File Layout 9-3
Design Assumptions 9-6
otbdlord (Download Summary of Outstanding Orders on OTB by Subclass) 9-6
Schedule 9-7
Design Overview 9-7
Restart/Recovery 9-7
I/O Specification 9-7
Output File Layout 9-7
Design Assumptions 9-10
otbupld (Upload OTB Budget from Planning Systems) 9-10
Schedule 9-10
Design Overview 9-10
Restart/Recovery 9-10
I/O Specification 9-11
Input File Layout 9-11
Design Assumptions 9-13
otbprg (Purge Aged Open To Buy Data) 9-13
Schedule 9-13
Design Overview 9-13
Restart/Recovery 9-13
Design Assumptions 9-13
otb_purge_job (Purge Aged Open To Buy Data) 9-14
Design Overview 9-14
Scheduling Constraints 9-14
Restart/Recovery 9-14
Key Tables Affected 9-14
Design Assumptions 9-15

10 Future Cost

Future Cost Events 10-1
Future Cost Engine Run Type Configuration 10-2
Synchronous 10-2
Asynchronous 10-3
Batch 10-4
Future Cost Engine Concurrency Control 10-4
Future Cost Engine Error Handling 10-4

Future Cost Engine Threading/Chunking 10-4
Future Cost Process 10-5
Batch Design Summary 10-5
costeventprg (Purge Aged Cost Events) 10-6
Schedule 10-6
Design Overview 10-6
Restart/Recovery 10-6
Design Assumptions 10-6
cost_event_purge_job (Purge Aged Cost Events) 10-6
Design Overview 10-7
Scheduling Constraints 10-7
Restart/Recovery 10-7
Key Tables Affected 10-7
Design Assumptions 10-8
fcexec (Execute Batch Calculation/Recalculation of Future Cost Values) 10-8
Schedule 10-9
Design Overview 10-9
Restart/Recovery 10-9
Design Assumptions 10-9
fc_pricechg (Use Pending Price Changes to Drive Recalculation of Pricing Cost for some Franchise Item/Locations) 10-9
Schedule 10-9
Design Overview 10-9
Restart/Recovery 10-10
Key Tables Affected 10-10
Design Assumptions 10-10
fcthreadexec (Prepare Threads for Batch Calculation/Recalculation of Future Cost Values) 10-10
Schedule 10-10
Design Overview 10-10
Restart/Recovery 10-11
Design Assumptions 10-11
rms_oi_pricecostrefresh.ksh (Pricing Cost Refresh) 10-11
Schedule 10-11
Design Overview 10-11
Restart/Recovery 10-11
Key Tables Affected 10-11
Design Assumptions 10-12
rms_oi_wacvarrefresh.ksh (WAC Refresh) 10-12
Schedule 10-12
Design Overview 10-12
Restart/Recovery 10-12
Key Tables Affected 10-12
Design Assumptions 10-12

11 Invoice Matching

Batch Design Summary 11-1

edidlinv (Download of Invoice For Invoice Matching) 11-1
Schedule 11-1
Design Overview 11-2
Restart/Recovery 11-2
I/O Specification 11-2
Output File Layout 11-2
Design Assumptions 11-14
invclshp (Close Aged Shipments to Prevent them from Matching Open Invoices) 11-15
Schedule 11-15
Design Overview 11-15
Restart/Recovery 11-15
Design Assumptions 11-15
invc_ship_close_job (Close Aged Shipments to Prevent them from Matching Open Invoices) 11-15
Design Overview 11-15
Scheduling Constraints 11-16
Restart/Recovery 11-16
Key Tables Affected 11-16
invprg (Purge Aged Invoices) 11-17
Schedule 11-17
Design Overview 11-17
Restart/Recovery 11-17
Design Assumptions 11-17
invoice_purge_job (Purge Aged Invoices) 11-17
Design Overview 11-18
Scheduling Constraints 11-18
Restart/Recovery 11-18
Key Tables Affected 11-19

12 Replenishment

Replenishment Sub Processes 12-1
Establishment/Management of Replenishment Calculation Attributes 12-2
Calculation of Recommended Order Quantity (ROQ) 12-2
Build Transactions to Replenish Inventory 12-3
Cleanup Replenishment Data 12-4
Batch Design Summary 12-4
replsizeprofile (Update Replenishment Size Profile) 12-5
Schedule 12-6
Design Overview 12-6
Restart/Recovery 12-6
Key Tables Affected 12-6
Design Assumptions 12-6
rplatupd (Update Replenishment Calculation Attributes) 12-6
Schedule 12-6
Design Overview 12-7
Restart/Recovery 12-7
Key Tables Affected 12-7

Design Assumptions 12-8
rilmaint (Update Replenishment Calculation Attributes by Item/Loc) 12-8
Schedule 12-8
Design Overview 12-8
Restart/Recovery 12-8
Key Tables Affected 12-9
Design Assumptions 12-9
repladj (Recalculate Maximum Levels for Floating Point Replenishment) 12-9
Schedule 12-9
Design Overview 12-9
Restart/Recovery 12-10
Key Tables Affected 12-10
Design Assumptions 12-10
replroq.ksh (Calculate Net Inventory) 12-10
Schedule 12-10
Design Overview 12-10
Restart/Recovery 12-11
Key Tables Affected 12-11
Design Assumptions 12-11
batch_reqext.ksh (Multithreading Wrapper for reqext) 12-11
Schedule 12-12
Design Overview 12-12
Restart/Recovery 12-12
Key Tables Affected 12-12
Design Assumptions 12-12
reqext (ROQ Calculation and Distribution for Item/Locs Replenished from WH) 12-12
Schedule 12-13
Design Overview 12-13
Restart/Recovery 12-13
Key Tables Affected 12-13
Design Assumptions 12-14
rplext.ksh (ROQ Calculation and Distribution for Item/Locs Replenished from Supplier 12-14
Schedule 12-15
Design Overview 12-15
Restart/Recovery 12-15
Locking Strategy 12-15
Performance Considerations 12-15
Key Tables Affected 12-16
Design Assumptions 12-16
ibexpl (Determines Eligible Investment Buy Opportunities) 12-16
Schedule 12-16
Design Overview 12-16
Restart/Recovery 12-17
Key Tables Affected 12-17
Design Assumptions 12-18
buyer_wksht_purge_job (Purge Aged Buyer Worksheet Results) 12-18
Design Overview 12-18

Scheduling Constraints 12-18
Restart/Recovery 12-19
Key Tables Affected 12-19
Design Assumptions 12-19
ibcalc (Calculate ROQ for Profitable Investment Buys) 12-19
Schedule 12-19
Design Overview 12-19
Restart/Recovery 12-20
Key Tables Affected 12-20
Design Assumptions 12-20
rplbld (Build Replenishment Orders) 12-21
Schedule 12-21
Design Overview 12-21
Restart/Recovery 12-21
Key Tables Affected 12-22
Design Assumptions 12-22
supsplit (Split Replenishment Orders Among Suppliers) 12-23
Schedule 12-23
Design Overview 12-23
Restart/Recovery 12-23
Key Tables Affected 12-23
Design Assumptions 12-23
rplsplit (Truck Splitting Optimization for Replenishment) 12-23
Schedule 12-24
Design Overview 12-24
Restart/Recovery 12-24
Key Tables Affected 12-24
Design Assumptions 12-25
rplapprv (Approve Replenishment Orders) 12-26
Schedule 12-26
Design Overview 12-26
Restart/Recovery 12-27
Key Tables Affected 12-27
Design Assumptions 12-27
batch_rplapprvgtax.ksh (Update Replenishment Order Taxes) 12-27
Schedule 12-28
Design Overview 12-28
Restart/Recovery 12-28
Key Tables Affected 12-28
Design Assumptions 12-29
repl_wf_order_sync.ksh (Sync Replenishment Franchise Orders 12-29
Schedule 12-29
Design Overview 12-29
Restart/Recovery 12-29
Key Tables Affected 12-30
Design Assumptions 12-30
rplprg (Purge Aged Replenishment Results) 12-30

Schedule 12-31
Design Overview 12-31
Restart/Recovery 12-31
Key Tables Affected 12-31
Design Assumptions 12-31
replenishment_purge_job (Purge Aged Replenishment Results) 12-31
Design Overview 12-32
Scheduling Constraints 12-32
Restart/Recovery 12-32
Key Tables Affected 12-32
Design Assumptions 12-33
rplathistprg (Purge Replenishment Attribute History) 12-33
Schedule 12-33
Design Overview 12-33
Restart/Recovery 12-33
Key Tables Affected 12-33
Design Assumptions 12-33
rplprg_month (Purge Replenishment Results History by Month) 12-33
Schedule 12-34
Design Overview 12-34
Restart/Recovery 12-34
Key Tables Affected 12-34
Design Assumptions 12-34
repl_indctn_purge.ksh (Purge Scheduled Replenishment Induction Staging Tables) 12-35
Schedule 12-35
Design Overview 12-35
Restart/Recovery 12-35
Key Tables Affected 12-35
Design Assumptions 12-36
replindbatch.ksh (Upload Replenishment Induction Data) 12-36
Schedule 12-36
Design Overview 12-36
Restart/Recovery 12-36
Key Tables Affected 12-36
Design Assumptions 12-37
buyer_wksht_purge_job (Purge Aged Buyer Worksheet Results) 12-37
Design Overview 12-37
Scheduling Constraints 12-38
Restart/Recovery 12-38
Key Tables Affected 12-38
Design Assumptions 12-38
investment_buy_purge_job (Purge Aged Investment Buy Results) 12-38
Design Overview 12-39
Scheduling Constraints 12-39
Restart/Recovery 12-39
Key Tables Affected 12-39
Design Assumptions 12-40

store_orders_purge_job (Purge Aged Store Orders Results) 12-40
Design Overview 12-40
Scheduling Constraints 12-40
Restart/Recovery 12-40
Key Tables Affected 12-41
Design Assumptions 12-41

13 Inventory

Batch Design Summary 13-1
edidlprd (Download Sales and Stock On Hand From RMS to Suppliers) 13-1
Schedule 13-1
Design Overview 13-2
Restart/Recovery 13-2
Key Tables Affected 13-2
I/O Specification 13-2
Output File Layout 13-3
Design Assumptions 13-6
ordinvupld (Upload and Process Inventory Reservations from ReSA) 13-6
Schedule 13-6
Design Overview 13-6
Restart/Recovery 13-7
Key Tables Affected 13-7
I/O Specification 13-7
Input File Layout 13-7
Design Assumptions 13-10
wasteadj (Adjust Inventory for Wastage Items) 13-10
Schedule 13-11
Design Overview 13-11
Restart/Recovery 13-11
Design Assumptions 13-11
refeodinventory (Refresh End of Day Inventory Snapshot) 13-11
Schedule 13-11
Design Overview 13-12
Restart/Recovery 13-12
Key Tables Affected 13-12
Design Assumptions 13-12
invaprg (Purge Aged Inventory Adjustments) 13-12
Schedule 13-12
Design Overview 13-13
Restart/Recovery 13-13
Key Tables Affected 13-13
Design Assumptions 13-13
inv_adj_purge_job (Purge Aged Inventory Adjustments) 13-13
Design Overview 13-13
Scheduling Constraints 13-14
Restart/Recovery 13-14
Key Tables Affected 13-14

Design Assumptions 13-14
customer_order_purge.ksh (Purge Aged Customer Orders) 13-14
Schedule 13-15
Design Overview 13-15
Restart/Recovery 13-15
Key Tables Affected 13-15
Design Assumptions 13-15
customer_orders_purge_job (Purge Aged Customer Orders) 13-15
Design Overview 13-15
Scheduling Constraints 13-16
Restart/Recovery 13-16
Key Tables Affected 13-16
Security Considerations 13-16

14 Transfers, Allocation, and RTV

Batch Design Summary 14-1
docclose (Close Transactions with no Expected Appointments, Shipments or Receipts) 14-2
Schedule 14-2
Design Overview 14-2
Restart/Recovery 14-2
Design Assumptions 14-2
doc_queue_close_job (Close Transactions with no Expected Appointments, Shipments or Receipts) 14-2
Design Overview 14-3
Scheduling Constraints 14-3
Restart/Recovery 14-3
Key Tables Affected 14-3
Design Assumptions 14-4
dummyctn (Reconcile Received Dummy Carton IDs with Expected Cartons) 14-4
Schedule 14-4
Design Overview 14-4
Restart/Recovery 14-4
Key Tables Affected 14-5
Design Assumptions 14-5
tamperctn (Detail Receive Damaged or Tampered with Cartons) 14-5
Schedule 14-6
Design Overview 14-6
Restart/Recovery 14-6
Key Tables Affected 14-6
Design Assumptions 14-6
distropcpub (Stage Regular Price Changes on Open Allocations/Transfers so Publishing Sends New Retail to Subscribing Applications) 14-6
Schedule 14-7
Design Overview 14-7
Restart/Recovery 14-7
Key Tables Affected 14-7
I/O Specification 14-8

Design Assumptions 14-8
mrt (Create Transfers for Mass Return Transfer) 14-8
Schedule 14-8
Design Overview 14-8
Restart/Recovery 14-8
Key Tables Affected 14-9
Design Assumptions 14-9
mrtrtv (Create Return to Vendor for Mass Return Transfer) 14-9
Schedule 14-9
Design Overview 14-9
Restart/Recovery 14-10
Key Tables Affected 14-10
Design Assumptions 14-10
mrtupd (Close Mass Return Transfers) 14-10
Schedule 14-10
Design Overview 14-10
Restart/Recovery 14-10
Key Tables Affected 14-11
Design Assumptions 14-11
mrtprg (Purge Aged Mass Return Transfers and RTV) 14-11
Schedule 14-11
Design Overview 14-11
Restart/Recovery 14-11
Key Tables Affected 14-12
Design Assumptions 14-12
mrt_purge_job (Purge Aged Mass Return Transfers and RTV) 14-12
Design Overview 14-12
Scheduling Constraints 14-13
Restart/Recovery 14-13
Key Tables Affected 14-13
Design Assumptions 14-14
rtvprg (Purge Aged Returns to Vendors) 14-14
Schedule 14-14
Design Overview 14-14
Restart/Recovery 14-14
Key Tables Affected 14-15
Design Assumptions 14-15
rtv_purge_job (Purge Aged Returns to Vendors) 14-15
Design Overview 14-15
Scheduling Constraints 14-16
Restart/Recovery 14-16
Key Tables Affected 14-16
Design Assumptions 14-17
tsfclose (Close Overdue Transfers) 14-17
Schedule 14-17
Design Overview 14-17
Restart/Recovery 14-17

Key Tables Affected 14-18
Design Assumptions 14-18
transfer_close_job (Close Overdue Transfers) 14-18
Design Overview 14-18
Scheduling Constraints 14-19
Restart/Recovery 14-19
Key Tables Affected 14-19
Design Assumptions 14-19
tsfprg (Purge Aged Transfers) 14-20
Schedule 14-20
Design Overview 14-20
Restart/Recovery 14-20
Key Tables Affected 14-20
Design Assumptions 14-20
transfer_purge_job (Purge Aged Transfers) 14-21
Design Overview 14-21
Scheduling Constraints 14-21
Restart/Recovery 14-21
Key Tables Affected 14-22
Design Assumptions 14-23
allocbt (Create Book Transfers for Allocations Between Warehouses in the Same Physical Warehouse) 14-23
Schedule 14-23
Design Overview 14-24
Restart/Recovery 14-24
Key Tables Affected 14-25
Design Assumptions 14-25

15 Sales Posting

Creating a POSU File 15-1
Sales Posting Business Process 15-1
Batch Design Summary 15-2
uploadsales.ksh (Upload POSU File for Processing) 15-2
Schedule 15-3
Design Overview 15-3
Restart/Recovery 15-3
Locking Strategy 15-3
Security Considerations 15-3
Performance Considerations 15-3
Security Considerations 15-3
I/O Specification 15-3
Input File Layout 15-3
Design Assumptions 15-8
Rolling up transactions to the item/store/price point 15-8
Reject File 15-9
uploadsales_all.ksh (Process Multiple POSU Files) 15-9
Schedule 15-10

Design Overview 15-10
Restart/Recovery 15-10
Locking Strategy 15-10
Security Considerations 15-10
Performance Considerations 15-10
Security Considerations 15-10
I/O Specification 15-10
Input File Layout 15-10
salesprocess.ksh (Main Processing of Staged Sale/Return Transactions) 15-10
Schedule 15-11
Design Overview 15-11
POSU Chunking 15-11
Restart/Recovery 15-12
Locking Strategy 15-12
Security Considerations 15-12
Performance Considerations 15-12
I/O Specification 15-13
Design Assumptions 15-13
Financial Transactions 15-13
salesgenrej.ksh (Reject POSU Transactions) 15-14
Schedule 15-14
Design Overview 15-14
Restart/Recovery 15-14
Performance Considerations 15-14
Reject File: 15-15
salesuploadarch.ksh (Archive Successfully Posted Transactions) 15-15
Schedule 15-15
Design Overview 15-15
Performance Considerations 15-15
Design Assumptions 15-15
salesuploadpurge.ksh (Purge Aged Archived POSU Transactions) 15-15
Schedule 15-16
Design Overview 15-16
Performance Considerations 15-16
Design Assumptions 15-16
sales_reprocess.ksh (Re-processing of Sale/Return Transactions Due to Chunk Not Process Issue) 15-16
Design Overview 15-16
Scheduling Constraints 15-16
Restart/Recovery 15-17
Locking Strategy 15-17
Security Considerations 15-17
Performance Considerations 15-17
Key Tables Affected 15-17
Integration Contract 15-17
Design Assumptions 15-17

file_upload_errors_purge.ksh (Purge FILE_UPLOAD_STATUS and FILE_UPLOAD_ERRORS Tables) 15-18
Schedule 15-18
Design Overview 15-18
Restart/Recovery 15-18
I/O Specification 15-19
Design Assumptions 15-19

16 Sales History

Batch Design Summary 16-1
hstbld (Weekly Sales History Rollup by Department, Class, and Subclass) 16-1
Schedule 16-2
Design Overview 16-2
Restart/Recovery 16-2
Design Assumptions 16-2
hstbld_diff (Weekly Sales History Rollup by Diff) 16-2
Schedule 16-2
Design Overview 16-2
Restart/Recovery 16-3
Design Assumptions 16-3
Key Tables Affected 16-3
hstbldmth (Monthly Sales History Rollup By Department, Class And Subclass) 16-3
Schedule 16-3
Design Overview 16-3
Restart/Recovery 16-4
Design Assumptions 16-4
hstbldmth_diff (Monthly Sales History Rollup By Diffs) 16-4
Schedule 16-4
Design Overview 16-4
Restart/Recovery 16-4
Locking Strategy 16-4
Design Assumptions 16-5
hstmthupd (Monthly Stock on Hand, Retail and Average Cost Values Update) 16-5
Schedule 16-5
Design Overview 16-5
Restart/Recovery 16-5
I/O Specification 16-5
hstwkupd (Weekly Stock on Hand and Retail Value Update for Item/Location) 16-5
Schedule 16-6
Design Overview 16-6
Restart/Recovery 16-6
I/O Specification 16-6
Design Assumptions 16-6
hstprg (Purge Aged Sales History) 16-6
Schedule 16-7
Design Overview 16-7
Restart/Recovery 16-7

Design Assumptions 16-7
history_purge_job (Purge Aged Sales History) 16-7
Design Overview 16-7
Scheduling Constraints 16-8
Restart/Recovery 16-8
Key Tables Affected 16-8
hstprg_diff (Purge Aged Sales History by Diff) 16-8
Schedule 16-9
Design Overview 16-9
Restart/Recovery 16-9
Design Assumptions 16-9
hist_diff_purge_job (Purge Aged Sales History by Diff) 16-9
Design Overview 16-9
Scheduling Constraints 16-10
Restart/Recovery 16-10
Key Tables Affected 16-10

17 Stock Count

Batch Design Summary 17-1
lifstkup (Conversion of RWMS Stock Count Results File to RMS Integration Contract) 17-2
Schedule 17-2
Design Overview 17-2
Restart/Recovery 17-2
Key Tables Affected 17-2
I/O Specification 17-2
Input File Layout 17-3
Output File Layout 17-4
Design Assumptions 17-5
stockcountupload.ksh (Upload Stock Count Results from Stores/Warehouses) 17-6
Schedule 17-6
Design Overview 17-6
Key Tables Affected 17-6
Input/Out Specification 17-6
Input File Layout 17-7
Design Assumptions 17-8
stkdly (Calculate Actual Current Shrinkage and Budgeted Shrink to Apply to Stock Ledger) 17-8
Schedule 17-9
Design Overview 17-9
Restart/Recovery 17-9
Key Tables Affected 17-9
Design Assumptions 17-10
stkprg (Purge Aged Stock Count) 17-10
Schedule 17-10
Design Overview 17-10
Restart/Recovery 17-10
Key Tables Affected 17-10

Design Assumption 17-11
stock_count_purge_job (Purge Aged Stock Count) 17-11
Design Overview 17-11
Scheduling Constraints 17-11
Restart/Recovery 17-12
Key Tables Affected 17-12
Design Assumption 17-12
stkschedxpld (Create Stock Count Requests Based on Schedules) 17-12
Schedule 17-13
Design Overview 17-13
Restart/Recovery 17-13
Key Tables Affected 17-13
Design Assumption 17-14
stake_sched_explode_job (Create Stock Count Requests Based on Schedules) 17-14
Design Overview 17-14
Scheduling Constraints 17-15
Restart/Recovery 17-15
Key Tables Affected 17-15
Design Assumption 17-16
stkupd (Stock Count Snapshot Update) 17-16
Schedule 17-16
Design Overview 17-16
Restart/Recovery 17-16
Key Tables Affected 17-16
Design Assumption 17-17
stkvar (Update Stock On Hand Based on Stock Count Results) 17-17
Schedule 17-17
Design Overview 17-17
Restart/Recovery 17-17
Key Tables Affected 17-18
Design Assumption 17-18
stkxpld (Explode Stock Count Requests to Item Level) 17-18
Schedule 17-19
Design Overview 17-19
Restart/Recovery 17-19
Key Tables Affected 17-19
Design Assumption 17-19
stockcountprocess.ksh (Process Stock Count Results) 17-20
Schedule 17-20
Design Overview 17-20
Restart/Recovery 17-20
Key Tables Affected 17-20
Design Assumption 17-21

18 Oracle Retail Trade Management

Simplified Trade Management Configuration 18-1
Simplified Trade Management Batch Program Notes 18-2

Batch Design Summary 18-2
cednld (Download of Customs Entry Transactions to Brokers) 18-3
Schedule 18-3
Design Overview 18-3
Restart/Recovery 18-3
Key Tables Affected 18-3
I/O Specification 18-4
Output File Layout 18-4
Design Assumptions 18-13
htsupld (Harmonized Tariff Schedule Upload) 18-13
Schedule 18-13
Design Overview 18-13
Restart/Recovery 18-13
Key Tables Affected 18-14
I/O Specification 18-15
Input File Layout 18-15
Original Input File 18-16
Design Assumptions 18-30
tranupld (Transportation Upload) 18-30
Schedule 18-30
Design Overview 18-31
Restart/Recovery 18-31
I/O Specification 18-31
Input File Layout 18-31
Design Assumptions 18-37
lcadnld (Letter of Credit Application Download) 18-37
Schedule 18-37
Design Overview 18-37
Restart/Recovery 18-37
I/O Specification 18-37
Output File Layout 18-38
lcmt700 (SWIFT File Conversion - Letter of Credit Application) 18-53
Schedule 18-53
Design Overview 18-53
I/O Specification 18-53
Output 18-53
lcupld (Letter of Credit Confirmation Upload) 18-54
Schedule 18-54
Design Overview 18-54
Restart/Recovery 18-54
I/O Specification 18-55
Input File Layout 18-55
lcmt730 (SWIFT File Conversion - Letter of Credit Confirmation) 18-56
Schedule 18-57
Design Overview 18-57
I/O Specification 18-57
Input File Layout 18-57

Output File Layout 18-61
Design Assumptions 18-63
lcmdnld (Letter of Credit Amendment Download) 18-63
Schedule 18-63
Design Overview 18-63
Restart/Recovery 18-63
I/O Specification 18-63
Output File Layout 18-64
lcmt707 (SWIFT File Conversion – Letter of Credit Amendment) 18-69
Schedule 18-69
Design Overview 18-69
I/O Specification 18-70
Output 18-70
lcup798 (Letter of Credit Drawdowns and Charges) 18-72
Schedule 18-72
Design Overview 18-72
Restart/Recovery 18-72
I/O Specification 18-72
Input File Layout 18-73
lcmt798 (SWIFT File Conversion – Letter of Credit Charges and Drawdowns) 18-75
Schedule 18-75
Design Overview 18-75
I/O Specification 18-75
Input File Layout 18-76
I/O Specification 18-82
Output File Layout 18-82

19 Stock Ledger

Stock Ledger Set Up and Accounting Methods 19-1
Process Flow 19-2
Batch Design Summary 19-2
salstage (Stage Stock Ledger Transactions for Additional Processing) 19-3
Schedule 19-3
Design Overview 19-3
Restart/Recovery 19-4
Design Assumptions 19-4
salapnd (Append Stock Ledger Information to History Tables) 19-4
Schedule 19-4
Design Overview 19-4
Restart/Recovery 19-4
Design Assumptions 19-4
saldly (Daily Rollup of Transaction Data for Stock Ledger) 19-4
Schedule 19-5
Design Overview 19-5
Restart/Recovery 19-5
Design Assumption 19-5
salweek (Weekly Rollup of Data/Calculations for Stock Ledger) 19-5

Schedule ... 19-6
Design Overview ... 19-6
Restart/Recovery ... 19-6
Design Assumptions ... 19-6
salmth (Monthly Rollup of Data/Calculations for Stock Ledger) ... 19-6
Schedule ... 19-7
Design Overview ... 19-7
Restart/Recovery ... 19-7
Design Assumptions ... 19-7
salmaint (Stock Ledger Table Maintenance) ... 19-7
Schedule ... 19-8
Design Overview ... 19-8
Restart/Recovery ... 19-8
Locking Strategy ... 19-8
Security Considerations ... 19-8
Performance Considerations ... 19-8
I/O Specification ... 19-8
stock_ledger_purge_job (Stock Ledger Table Maintenance) ... 19-8
Design Overview ... 19-8
Scheduling Constraints ... 19-9
Restart/Recovery ... 19-9
Locking Strategy ... 19-9
Security Considerations ... 19-9
Performance Considerations ... 19-9
Key Tables Affected ... 19-9
I/O Specification ... 19-10
saleoh (End Of Half Rollup of Data/Calculations for Stock Ledger) ... 19-10
Schedule ... 19-10
Design Overview ... 19-10
Restart/Recovery ... 19-10
Design Assumptions ... 19-11
salprg (Purge Stock Ledger History) ... 19-11
Schedule ... 19-11
Design Overview ... 19-11
Restart/Recovery ... 19-11
Design Assumptions ... 19-11
stkledgr_hist_purge_job (Purge Stock Ledger History) ... 19-11
Design Overview ... 19-12
Scheduling Constraints ... 19-12
Restart/Recovery ... 19-12
Key Tables Affected ... 19-12
Design Assumptions ... 19-13
nwppurge (Purge of Aged End of Year Inventory Positions) ... 19-13
Schedule ... 19-13
Design Overview ... 19-13
Restart/Recovery ... 19-13
Design Assumptions ... 19-13

nwp_purge_job (Purge of Aged End of Year Inventory Positions) 19-13
Design Overview 19-14
Scheduling Constraints 19-14
Restart/Recovery 19-14
Key Tables Affected 19-14
Design Assumptions 19-15
nwpyearend (End of Year Inventory Position Snapshot) 19-15
Schedule 19-15
Design Overview 19-15
Restart/Recovery 19-15
Design Assumptions 19-16
stlgdnld (Weekly or Historical Download of Stock Ledger Data) 19-16
Schedule 19-16
Design Overview 19-16
Restart/Recovery 19-16
I/O Specification 19-16
Input File Layout 19-17
Output File Layout 19-17
Design Assumptions 19-20
otbdlsal (Open To Buy Download Stock Ledger) 19-20
Schedule 19-21
Design Overview 19-21
Restart/Recovery 19-21
Locking Strategy 19-21
Security Considerations 19-21
Performance Considerations 19-21
I/O Specification 19-21
Output File Format 19-22
Design Assumptions 19-31
trandataload.ksh (External Transaction Data Upload) 19-31
Schedule 19-31
Design Overview 19-31
Chunking Logic 19-32
Restart/Recovery 19-33
I/O Specification - Input File Specification 19-33
File Layout 19-33
Design Assumptions 19-34
trandataprocess.ksh (External Transaction Data Process) 19-34
Schedule 19-34
Design Overview 19-34
Restart/Recovery 19-36
Design Assumptions 19-36

20 Franchise Management

Customers 20-1
Costing 20-1
Franchise Orders 20-2

Franchise Returns 20-2
Batch Design Summary 20-2
fcosttmplupld (Upload Cost Buildup Template) 20-3
Schedule 20-3
Design Overview 20-3
Restart/Recovery 20-3
Key Tables Affected 20-4
I/O Specification 20-5
SQL Loader Input File Layout 20-5
Design Assumptions 20-9
fcosttmplprocess (Process Cost Buildup Template Upload) 20-9
Schedule 20-10
Design Overview 20-10
Restart/Recovery 20-10
Key Tables Affected 20-10
Design Assumptions 20-11
fcosttmplpurge (Purge Staged Cost Template Data) 20-11
Schedule 20-11
Design Overview 20-11
Restart/Recovery 20-11
Key Tables Affected 20-12
Design Assumptions 20-12
wf_cost_template_purge_job (Purge Staged Cost Template Data) 20-12
Design Overview 20-12
Scheduling Constraints 20-12
Restart/Recovery 20-13
Key Tables Affected 20-13
Design Assumptions 20-13
fcustomerupload (Franchise Customer Upload) 20-13
Schedule 20-14
Design Overview 20-14
Restart/Recovery 20-14
Key Tables Affected 20-14
I/O Specification 20-15
Input File Layout 20-15
Design Assumptions 20-18
fcustomerprocess (Process Uploaded Franchise Customers and Customer Groups) 20-18
Schedule 20-19
Design Overview 20-19
Restart/Recovery 20-19
Commit Points 20-19
Key Tables Affected 20-19
Design Assumptions 20-20
Program Flow 20-20
fcustupldpurge (Franchise Customer Staging Purge) 20-20
Schedule 20-20
Design Overview 20-20

Restart/Recovery 20-21
Key Tables Affected 20-21
Design Assumptions 20-21
wfordupld.ksh (Franchise Order Upload) 20-21
Schedule 20-21
Design Overview 20-21
Restart/Recovery 20-22
Key Tables Affected 20-22
I/O Specification 20-23
SQL Loader Input File Layout 20-24
Design Assumptions 20-26
wf_apply_supp_cc.ksh (Apply Supplier Cost Change to Franchise Orders) 20-26
Schedule 20-26
Design Overview 20-26
Restart/Recovery 20-26
Key Tables Affected 20-27
Design Assumptions 20-27
wfordcls (Franchise Order Close) 20-27
Schedule 20-27
Design Overview 20-27
Restart/Recovery 20-28
Key Tables Affected 20-28
Design Assumptions 20-28
wf_orders_close_job (Franchise Order Close) 20-28
Design Overview 20-28
Scheduling Constraints 20-29
Restart/Recovery 20-29
Key Tables Affected 20-29
Design Assumptions 20-30
wfordprg (Franchise Order Purge) 20-30
Schedule 20-30
Design Overview 20-30
Restart/Recovery 20-30
Key Tables Affected 20-30
Design Assumptions 20-31
wf_orders_purge_job (Franchise Order Purge) 20-31
Design Overview 20-31
Scheduling Constraints 20-32
Restart/Recovery 20-32
Key Tables Affected 20-32
Design Assumptions 20-33
wfretupld.ksh (Franchise Return Upload) 20-33
Schedule 20-33
Design Overview 20-33
Restart/Recovery 20-33
Key Tables Affected 20-34
I/O Specification 20-34

SQL Loader Input File Layout 20-34
Design Assumptions 20-37
wfretcls (Franchise Return Close) 20-37
Schedule 20-37
Design Overview 20-37
Restart/Recovery 20-37
Key Tables Affected 20-37
Design Assumptions 20-38
wf_returns_close_job (Franchise Return Close) 20-38
Design Overview 20-38
Scheduling Constraints 20-38
Restart/Recovery 20-39
Key Tables Affected 20-39
Design Assumptions 20-39
wfrtnprg (Franchise Return Purge) 20-39
Schedule 20-39
Design Overview 20-39
Restart/Recovery 20-40
Key Tables Affected 20-40
Design Assumptions 20-40
wf_returns_purge_job (Franchise Return Purge) 20-40
Design Overview 20-40
Scheduling Constraints 20-41
Restart/Recovery 20-41
Key Tables Affected 20-41
Design Assumptions 20-41
wfslsupld.ksh (Upload of Franchise Sales to Merchandising) 20-42
Schedule 20-42
Design Overview 20-42
Restart/Recovery 20-42
I/O Specification 20-43
Input File Layout 20-43
Design Assumptions 20-44
wfbillex.ksh (Franchise Billing Extract) 20-44
Schedule 20-45
Design Overview 20-45
Restart/Recovery 20-45
I/O Specification 20-45
Output File Layout 20-45
Design Assumptions 20-48

21 Competitive Pricing

Batch Design Summary 21-1
cmpupld (Upload Competitor's Prices) 21-1
Schedule 21-1
Design Overview 21-1
Restart/Recovery 21-2

Key Tables Affected 21-2
I/O Specification 21-2
Input File Layout 21-2
Design Assumptions 21-6
cmpprg.pc (Purge Aged Competitive Pricing Data) 21-6
Schedule 21-6
Design Overview 21-6
Restart/Recovery 21-6
Key Tables Affected 21-6
Design Assumptions 21-6
comp_pricing_purge_job (Purge Aged Competitive Pricing Data) 21-7
Design Overview 21-7
Scheduling Constraints 21-7
Restart/Recovery 21-7
Key Tables Affected 21-8
Design Assumptions 21-8

22 Item Induction

Batch Design Summary 22-2
loadods.ksh (Item Induction) 22-2
Schedule 22-2
Design Overview 22-2
Restart/Recovery 22-3
I/O Specification 22-3
Input File Specification - SQL Loader Input File Layout 22-3
iindbatch.ksh (Upload Item Data) 22-3
Schedule 22-3
Design Overview 22-3
Restart/Recovery 22-4
Key Tables Affected 22-4
Design Assumptions 22-6
ld_iindfiles.ksh (Upload Data From Templates) 22-6
Schedule 22-6
Design Overview 22-6
Restart/Recovery 22-6
Key Tables Affected 22-6
Design Assumptions 22-7
itm_indctn_purge (Purge Item Induction Staging Tables) 22-7
Schedule 22-7
Design Overview 22-7
Restart/Recovery 22-7
Key Tables Affected 22-8
Design Assumptions 22-9

23 Integration with Point of Sale

Bulk Export Pattern 23-1
Program Summary 23-2

export_merchhier.ksh (Extract of Merchandise Hierarchy data) ... 23-3
Schedule ... 23-3
Design Overview ... 23-3
Restart/Recovery ... 23-4
Key Tables Affected ... 23-4
I/O Specification ... 23-4
Design Assumptions ... 23-4
export_orghier.ksh (Extract of Organizational Hierarchy Data) ... 23-4
Schedule ... 23-4
Design Overview ... 23-5
Restart/Recovery ... 23-5
Key Tables Affected ... 23-5
I/O Specification ... 23-5
Design Assumptions ... 23-5
export_stores.ksh (Extract of Store Data) ... 23-5
Schedule ... 23-6
Design Overview ... 23-6
Restart/Recovery ... 23-6
Key Tables Affected ... 23-6
I/O Specification ... 23-6
Design Assumptions ... 23-7
export_diffs.ksh (Extraction of differentiators data defined for a differentiator type) ... 23-7
Schedule ... 23-7
Design Overview ... 23-7
Restart/Recovery ... 23-7
Key Tables Affected ... 23-7
I/O Specification ... 23-8
Design Assumptions ... 23-8
export_diffgrp.ksh (Extraction of differentiator groups data) ... 23-8
Schedule ... 23-8
Design Overview ... 23-8
Restart/Recovery ... 23-8
Key Tables Affected ... 23-8
I/O Specification ... 23-9
Design Assumptions ... 23-9
export_itemloc.ksh (Extraction of item location data) ... 23-9
Schedule ... 23-9
Design Overview ... 23-9
Restart/Recovery ... 23-10
Key Tables Affected ... 23-10
I/O Specification ... 23-10
Design Assumptions ... 23-10
export_itemvat.ksh (Extraction of vat item data) ... 23-11
Schedule ... 23-11
Design Overview ... 23-11
Restart/Recovery ... 23-11
Key Tables Affected ... 23-12

I/O Specification 23-12
Design Assumptions 23-12
export_itemmaster.ksh (Extraction of item data) 23-12
Schedule 23-12
Design Overview 23-12
Restart/Recovery 23-13
Key Tables Affected 23-13
I/O Specification 23-13
Design Assumptions 23-14
export_vat.ksh (Extraction of vat data) 23-14
Schedule 23-14
Design Overview 23-14
Restart/Recovery 23-14
Key Tables Affected 23-14
I/O Specification 23-15
Design Assumptions 23-15
export_relitem.ksh (Extraction of item data) 23-15
Schedule 23-15
Design Overview 23-15
Restart/Recovery 23-16
Key Tables Affected 23-16
I/O Specification 23-16
Design Assumptions 23-16
taxdnld (Tax Download to 3rd Party POS in Global Tax [GTAX] Implementations) 23-16
Schedule 23-17
Design Overview 23-17
Restart/Recovery 23-17
Key Tables Affected 23-17
Integration Contract 23-17
Output File Layout 23-18
Design Assumptions 23-19
poscdnld (Download of POS Configuration Data to 3rd Party POS) 23-19
Schedule 23-19
Design Overview 23-19
Restart/Recovery 23-20
Key Tables Affected 23-20
I/O Specification 23-20
Output File Layout 23-21
Design Assumptions 23-26
export_stg_purge.ksh (Purging of all the extracted data) 23-26
Schedule 23-27
Design Overview 23-27
Restart/Recovery 23-27
Key Tables Affected 23-27
Design Assumptions 23-28

24 Integration with Advanced Inventory Planning

Foundation Data vs Transaction/Inventory Data 24-1
Program Summary 24-1
rmse_aip_batch (Optional Wrapper Script to run all AIP Extracts) 24-2
Schedule 24-3
Design Overview 24-3
Restart/Recovery 24-4
I/O Specification 24-4
pre_rmse_aip (Extract of Merchandising System level settings for AIP) 24-4
Schedule 24-4
Design Overview 24-4
Restart/Recovery 24-4
Key Tables Affected 24-4
I/O Specification 24-5
rmse_aip_merchhier (Extract of Merchandise Hierarchy for AIP) 24-8
Schedule 24-8
Design Overview 24-8
Restart/Recovery 24-8
I/O Specification 24-8
File Layout 24-8
rmse_aip_orghier.ksh (Extract of Organization Hierarchy for AIP) 24-9
Schedule 24-9
Design Overview 24-9
Restart/Recovery 24-9
Key Tables Affected 24-9
I/O Specification 24-10
Output File Layout 24-10
rmse_aip_item_master (RMS Extract of Items for AIP) 24-10
Schedule 24-10
Design Overview 24-11
Restart/Recovery 24-11
Locking Strategy 24-11
Security Considerations 24-11
Performance Considerations 24-11
I/O Specification 24-11
Output File Layout 24-11
Integration Contract 24-12
Output File Layout 24-13
rmse_aip_store (Extract of Stores for AIP) 24-13
Schedule 24-13
Design Overview 24-13
Restart/Recovery 24-13
Key Tables Affected 24-13
I/O Specification 24-13
Output File Layout 24-14
Design Assumptions 24-14
rmse_aip_wh (Extract of Warehouses for AIP) 24-14

Schedule 24-14
Design Overview 24-14
Restart/Recovery 24-15
Key Tables Affected 24-15
I/O Specification 24-15
Output File Layout 24-15
I/O Specification 24-15
Output File Layout 24-15
I/O Specification 24-16
Output File Layout 24-16
Design Assumptions 24-16
rmse_aip_substitute_items (Extract of Substitute Items for AIP) 24-16
Schedule 24-16
Design Overview 24-16
Restart/Recovery 24-17
Key Tables Affected 24-17
I/O Specification 24-17
Output File Layout 24-17
Design Assumptions 24-17
rmse_aip_suppliers (Extract of Suppliers for AIP) 24-17
Schedule 24-18
Design Overview 24-18
Restart/Recovery 24-18
Key Tables Affected 24-18
I/O Specification 24-18
Output File Layout 24-18
I/O Specification 24-18
Output File Layout 24-19
I/O Specification 24-19
Output File Layout 24-19
Design Assumptions 24-19
rmse_aip_alloc_in_well (Extract of Allocations in the Well Quantities for AIP) 24-19
Schedule 24-19
Design Overview 24-20
Restart/Recovery 24-20
I/O Specification 24-20
File Layout 24-20
rmse_aip_cl_po (Extract of AIP Generated POs, Allocations and Transfers Cancelled or Closed in Merchandising for AIP) 24-22
Schedule 24-22
Design Overview 24-22
Restart/Recovery 24-23
I/O Specification 24-23
Output File Layout 24-23
rmse_aip_future_delivery_alloc (Extract of Allocation Quantities for Future Delivery for AIP) 24-23
Schedule 24-23
Design Overview 24-23

Restart/Recovery 24-24
I/O Specification 24-24
Output File Layout 24-24
rmse_aip_future_delivery_order (Extract of Purchase Order Quantities for Future Delivery to AIP) 24-27
Schedule 24-27
Design Overview 24-27
Restart/Recovery 24-27
I/O Specification 24-28
File Layout 24-28
rmse_aip_future_delivery_tsf (Extract On Order and In Transit Transfer Quantities for Future Delivery for AIP) 24-29
Schedule 24-30
Design Overview 24-30
Restart/Recovery 24-30
I/O Integration 24-30
Output File Layout 24-30
rmse_aip_item_loc_traits (Extract of Shelf Life on Receipt Location Trait for AIP) 24-34
Schedule 24-34
Design Overview 24-34
Restart/Recovery 24-34
I/O Specification 24-34
Output File Layout 24-34
rmse_aip_item_retail (Extract of Forecasted Items for AIP) 24-35
Schedule 24-35
Design Overview 24-35
Non-Pack Items 24-35
Simple Pack Components 24-35
Restart/Recovery 24-35
I/O Specification 24-35
Output File Layout 24-36
rmse_aip_item_sale (Extract of Scheduled Item Maintenance On/Off Sale Information for AIP) .. 24-36
Schedule 24-36
Design Overview 24-37
Restart/Recovery 24-37
I/O Specification 24-37
Output File Layout 24-37
Integration Contract 24-38
Output File Layout 24-38
File Layout 24-38
rmse_aip_item_supp_country (Extract of Order Multiples by Item/Supplier/Origin Country for AIP) 24-39
Schedule 24-39
Design Overview 24-39
Restart/Recovery 24-39
I/O Specification 24-39
File Layout 24-39

Integration Contract 24-40
File Layout 24-40
File Layout 24-41
rmse_aip_rec_qty (Extract of Received PO, Allocation and Transfer Quantities for AIP) 24-41
Schedule 24-41
Design Overview 24-41
Restart/Recovery 24-42
Key Tables Affected 24-42
I/O Specification 24-42
Output File Layout 24-43
Design Assumptions 24-43
rmse_aip_store_cur_inventory (Extract of Store Current Inventory data for AIP) 24-43
Schedule 24-44
Design Overview 24-44
Restart/Recovery 24-44
Key Tables Affected 24-44
I/O Specification 24-44
Output File Layout 24-45
Design Assumptions 24-45
rmse_aip_tsf_in_well (Extract of Transfer in the Well Quantities to AIP) 24-45
Schedule 24-45
Design Overview 24-45
Restart/Recovery 24-45
Key Tables Affected 24-46
I/O Specification 24-46
Output File Layout 24-46
Design Assumptions 24-48
rmse_aip_wh_cur_inventory (Extract of Warehouse Current Inventory for AIP) 24-48
Schedule 24-48
Design Overview 24-48
Restart/Recovery 24-49
Key Tables Affected 24-49
I/O Specification 24-49
Output File Layout 24-49
Design Assumptions 24-50

25 Integration with General Ledger

Batch Design Summary 25-1
dealfinc (Calculation of Fixed Deal Income for General Ledger) 25-1
Schedule 25-1
Design Overview 25-2
Restart/Recovery 25-2
I/O Specification 25-2
Design Assumptions 25-2
fifgldn1 (Interface to General Ledger of Item/Loc Level Transactions) 25-2
Schedule 25-2
Design Overview 25-2

Restart/Recovery 25-3
I/O Specification 25-3
Design Assumptions 25-3
fifgldn2 (Interface to General Ledger of Rolled Up Transactions) 25-3
Schedule 25-3
Design Overview 25-3
Restart/Recovery 25-3
I/O Specification 25-4
Design Assumptions 25-4
fifgldn3 (Interface to General Ledger of Month Level Information) 25-4
Schedule 25-4
Design Overview 25-4
Restart/Recovery 25-4
I/O Specification 25-4
Design Assumptions 25-5
gl_extract.ksh (Extraction of General Ledger transaction data from Merchandising and Sales Audit) 25-5
Schedule 25-5
Design Overview 25-5
Restart/Recovery 25-5
I/O Specification 25-5
Output File Layout 25-5
Design Assumptions 25-7
Finance General Ledger to RFI (BDI_RFI_FinGenLdgr_Tx_PF_From_RMS_JOB) 25-7
Design Overview 25-7
Scheduling Constraints 25-8
Restart/Recovery 25-8
Key Tables Affected 25-8
Integration Contract 25-9

26 Integration with Oracle Retail Planning and Forecasting

Integration to Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS) 26-1
Integration to Oracle Retail Assortment and Item Planning for Fashion/ Softlines Cloud Service (APCS) 26-1
Integration from Oracle Retail Assortment and Item Planning for Fashion/ Softlines Cloud Service (APCS) 26-2
Integration to Oracle Retail Demand Forecasting Cloud Service (RDFCS) 26-3
Integration from Oracle Retail Demand Forecasting Cloud Service (RDFCS) 26-4
Data Maintenance 26-4
Integration Program Summary 26-4
Merchandise Hierarchy and Item Extract to Planning and Forecasting (BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB) 26-4
Design Overview 26-5
Scheduling Constraints 26-6
Restart/Recovery 26-6
Key Tables Affected 26-7
Integration Contract 26-7

Organizational Hierarchy Extract to Planning and Forecasting (BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB) 26-9
Design Overview ... 26-10
Scheduling Constraints ... 26-11
Restart/Recovery ... 26-11
Key Tables Affected ... 26-11
Integration Contract ... 26-12
Store Extract to Planning and Forecasting (BDI_RPAS_Store_Fnd_PF_From_RMS_JOB) 26-13
Design Overview ... 26-14
Scheduling Constraints ... 26-15
Restart/Recovery ... 26-15
Key Tables Affected ... 26-15
Integration Contract ... 26-16
Brand Extract to Planning (BDI_RPAS_Brand_Fnd_PF_From_RMS_JOB) ... 26-16
Design Overview ... 26-17
Scheduling Constraints ... 26-18
Restart/Recovery ... 26-18
Key Tables Affected ... 26-18
Integration Contract ... 26-18
Calendar Extract to Planning and Forecasting (BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB) 26-19
Design Overview ... 26-19
Scheduling Constraints ... 26-20
Restart/Recovery ... 26-21
Key Tables Affected ... 26-21
Integration Contract ... 26-21
Currency Rates Extract to Planning and Forecasting (BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB) 26-22
Design Overview ... 26-22
Scheduling Constraints ... 26-24
Restart/Recovery ... 26-24
Key Tables Affected ... 26-24
Integration Contract ... 26-24
Differentiator Extract to Planning (BDI_RPAS_Diff_Fnd_PF_From_RMS_JOB) ... 26-25
Design Overview ... 26-25
Scheduling Constraints ... 26-26
Restart/Recovery ... 26-26
Key Tables Affected ... 26-26
Integration Contract ... 26-27
Supplier Extract to Planning (BDI_RPAS_Supplier_Fnd_PF_From_RMS_JOB) ... 26-27
Design Overview ... 26-27
Scheduling Constraints ... 26-29
Restart/Recovery ... 26-29
Key Tables Affected ... 26-29
Integration Contract ... 26-29
UDA Extract to Planning (BDI_RPAS_UdaAndUdaValues_Fnd_PF_From_RMS_JOB) ... 26-29
Design Overview ... 26-30
Scheduling Constraints ... 26-31

Restart/Recovery 26-31
Key Tables Affected 26-31
Integration Contract 26-31
Inventory Extract to Planning (BDI_MFP_Inventory_Tx_PF_From_RMS_JOB) 26-32
Design Overview 26-32
Scheduling Constraints 26-34
Restart/Recovery 26-34
Key Tables Affected 26-34
Integration Contract 26-34
OnOrder Extract to Planning (BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB) 26-35
Design Overview 26-36
Scheduling Constraints 26-37
Restart/Recovery 26-37
Key Tables Affected 26-37
Integration Contract 26-38
Transaction Data Extract to Planning (BDI_MFP_TranData_Tx_PF_From_RMS_JOB) 26-38
Design Overview 26-38
Scheduling Constraints 26-40
Restart/Recovery 26-40
Key Tables Affected 26-40
Integration Contract 26-40
UDA Item Extract to Planning and Forecasting (BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB) 26-45
Design Overview 26-45
Scheduling Constraints 26-46
Restart/Recovery 26-47
Key Tables Affected 26-47
Integration Contract 26-47
Out of Stock Extract to Forecasting (BDI_RDF_StockOut_Tx_PF_From_RMS_JOB) 26-47
Design Overview 26-48
Scheduling Constraints 26-49
Restart/Recovery 26-49
Key Tables Affected 26-49
Integration Contract 26-49
Weekly Sales Extract to Forecasting (BDI_RDF_WeeklySales_Tx_PF_From_RMS_JOB) .. 26-50
Design Overview 26-50
Scheduling Constraints 26-51
Restart/Recovery 26-51
Key Tables Affected 26-52
Integration Contract 26-52
Upload to RMS 26-52
Weekly/Daily ItemForecast Upload (load_item_forecast) 26-52
Schedule 26-52
Design Overview 26-53
Restart/Recovery 26-53
Integration Contract 26-53
I/O Specification 26-54

Design Assumption 26-54
Data Maintenance 26-54
Retain Item Forecast History (rms_oi_forecast_history.ksh) 26-55
Schedule 26-55
Design Overview 26-55
Restart/Recovery 26-55
Key Tables Affected 26-55
Design Assumptions 26-55
Purge Forecast Data (fcstprg) 26-55
Schedule 26-56
Design Overview 26-56
Restart/Recovery 26-56
Key Tables Affected 26-56
Design Assumptions 26-56
Purge Forecast Data (forecast_data_purge_job) 26-56
Design Overview 26-56
Scheduling Constraints 26-57
Restart/Recovery 26-57
Key Tables Affected 26-57
Design Assumptions 26-57

27 Oracle Retail Sales Audit Batch Process and Designs

Oracle Retail Sales Audit Dataflow Diagram 27-1
Oracle Retail Sales Import Process 27-2
POS File Validation/Upload Sub-Process saimiptlog vs saimptlogi 27-3
Total Calculations and Rules 27-3
Totals when Transactions are Modified 27-4
Oracle Retail Sales Export Process 27-4
Full Disclosure and Post-export Changes 27-4
Batch Design Summary of Sales Audit Modules 27-5
Import Process Programs 27-5
Totals/Rules Programs 27-5
Export Programs 27-5
Other Programs 27-6
sastdycr (Create Store Day for Expected Transactions) 27-6
Schedule 27-6
Design Overview 27-6
Restart/Recovery 27-6
Design Assumptions 27-6
sagetref (Get Reference Data for Sales Audit Import Processing) 27-6
Schedule 27-7
Design Overview 27-7
Restart/Recovery 27-8
I/O Specification 27-8
File Name: Item File 27-8
File Name: Waste Data File 27-9
File Name: Reference Item Data 27-9

File Name: Primary Variant Data File ... 27-9
File Name: Variable Weight UPC Definition File ... 27-10
File Name: Valid Store/Day Combination File ... 27-10
File Name: Codes File ... 27-10
File Name: Error Information File ... 27-11
File Name: Store POS Mapping File ... 27-11
File Name: Tender Type Mapping File ... 27-11
File Name: Merchant Code Mapping File ... 27-11
File Name: Partner Mapping File ... 27-12
File Name: Supplier Mapping File ... 27-12
File Name: Employee Mapping File ... 27-12
File Name: Banner Information File ... 27-12
File Name: Currency Information File ... 27-13
File Name: Promotion Information File ... 27-13
File Name: Warehouse Information File ... 27-13
File Name: Inventory Status Information File ... 27-13
Design Assumptions ... 27-14
A Note about Primary Variant Relationships ... 27-14
saimptlog/saimptlogi (Import of Unaudited Transaction Data from POS to Sales Audit) ... 27-15
Schedule ... 27-15
Design Overview ... 27-15
Restart and Recovery ... 27-17
I/O Specification ... 27-17
Output File Layout ... 27-19
Control Files ... 27-21
Sales Audit Interface File Layout [rtlog] ... 27-32
Common Requirements/Validations ... 27-57
Requirements per Record Types ... 27-58
Code Type Validations ... 27-58
Transaction of Type SALE ... 27-62
Transaction of Type PVOID ... 27-64
Transaction of type RETURN ... 27-65
Transaction of type SPLORD ... 27-65
Transaction of type EEXCH ... 27-65
Transaction of type PAIDIN ... 27-66
Transaction Type PAIDOU ... 27-66
Transaction of Type PULL ... 27-67
Transaction of Type LOAN ... 27-67
Transaction Type Cond ... 27-68
Transaction of Type TOTAL ... 27-68
Transaction of Type METER ... 27-68
Transaction of Type PUMPT ... 27-69
Transactions of Type TANKDP ... 27-69
Transaction of Type DCLOSE ... 27-69
Transaction of Type REOPEN ... 27-70
Transaction of Type OTHER ... 27-70
Design Assumptions ... 27-70

DCLOSE Transaction Type ... 27-71
The Reopen Transaction Type ... 27-71
saimptlogtdup_upd (Processing to Allow Re-Upload of Deleted Transactions) ... 27-71
Schedule ... 27-72
Design Overview ... 27-72
Restart/Recovery ... 27-72
Design Assumptions ... 27-72
saimptlogfin (Complete Transaction Import Processing) ... 27-72
Schedule ... 27-72
Design Overview ... 27-72
Restart/Recovery ... 27-73
Design Assumptions ... 27-73
savouch (Sales Audit Voucher Upload) ... 27-73
Schedule ... 27-73
Design Overview ... 27-73
Restart/Recovery ... 27-73
I/O Specification ... 27-73
Input File Layout ... 27-74
Design Assumptions ... 27-77
saimpadj (Import Total Value Adjustments From External Systems to ReSA) ... 27-77
Schedule ... 27-77
Design Overview ... 27-77
Restart/Recovery ... 27-78
I/O Specification ... 27-78
Input File Layout ... 27-78
Design Assumptions ... 27-79
satotals (Calculate Totals based on Client Defined Rules) ... 27-80
Schedule ... 27-80
Design Overview ... 27-80
Restart/Recovery ... 27-80
Design Assumptions ... 27-80
sa_totals_calc_job (Calculate Totals based on Client Defined Rules) ... 27-80
Design Overview ... 27-81
Scheduling Constraints ... 27-81
Restart/Recovery ... 27-81
Key Tables Affected ... 27-82
Design Assumptions ... 27-82
sarules (Evaluate Transactions and Totals based on Client Defined Rules) ... 27-82
Schedule ... 27-82
Design Overview ... 27-83
Restart/Recovery ... 27-83
Design Assumptions ... 27-83
sa_rules_eval_job (Evaluate Transactions and Totals based on Client Defined Rules) ... 27-83
Design Overview ... 27-83
Scheduling Constraints ... 27-84
Restart/Recovery ... 27-84
Key Tables Affected ... 27-84

Design Assumptions 27-85
sapreexp (Prevent Duplicate Export of Total Values from ReSA) 27-85
Schedule 27-85
Design Overview 27-85
Restart/Recovery 27-85
Design Assumptions 27-85
saexprms (Export of POS transactions from Sales Audit to Merchandising) 27-86
Schedule 27-86
Design Overview 27-86
Restart/Recovery 27-86
I/O Specification 27-86
Design Assumptions 27-91
saordinvexp (Export Inventory Reservation/Release for In Store Customer Order & Layaway Transactions from ReSA) 27-91
Schedule 27-91
Design Overview 27-91
Restart/Recovery 27-92
I/O Specification 27-92
Output File Layout 27-92
Design Assumptions 27-94
saexpdw (Export from Sales Audit to Oracle Retail Analytics) 27-94
Schedule 27-94
Design Overview 27-95
Restart/Recovery 27-95
I/O Specification 27-95
Sales Audit - File Layout - Retail Analytics 27-96
RDWT File 27-96
Transaction Item Information Produced by saexpdw.pc after Translation by resa2dw 27-105
RDWF File 27-111
Retail Analytics Form of Payment File after Translation by resa2dw 27-113
RDWS File 27-115
Store Totals Information after Translation by resa2dw 27-117
RDWC File 27-117
Cashier/ Register Totals Information after Translation by resa2dw 27-119
Design Assumptions 27-120
saexpsim (Export of Revised Sale/Return Transactions from ReSA to SIM) 27-120
Schedule 27-121
Design Overview 27-121
Restart/Recovery 27-121
I/O Specification 27-121
Output File Layout 27-122
Design Assumptions 27-125
saexpim (Export DSD and Escheatment from Sales Audit to Invoice Matching) 27-125
Schedule 27-125
Design Overview 27-125
Restart/Recovery 27-126

Integration Contract 27-126
Design Assumptions 27-126
saexpgl (Post User Defined Totals from Sales Audit to General Ledger) 27-126
Schedule 27-127
Design Overview 27-127
Restart/Recovery 27-127
I/O Specification 27-127
Design Assumptions 27-127
ang_saplgen (Extract of POS Transactions by Store/Date from Sales Audit for Web Search) 27-127
Schedule 27-128
Design Overview 27-128
Restart/Recovery 27-128
I/O Specification 27-128
Output File Layout 27-128
Design Assumptions 27-129
saescheat (Download of Escheated Vouchers from Sales Audit for Payment) 27-129
Schedule 27-129
Design Overview 27-129
Restart/Recovery 27-130
I/O Specification 27-130
Design Assumptions 27-130
saescheat_nextesn (Generate Next Sequence for Escheatment Processing) 27-130
Schedule 27-130
Design Overview 27-130
Restart/Recovery 27-130
Design Assumptions 27-130
saexpach (Download from Sales Audit to Account Clearing House (ACH) System) 27-131
Schedule 27-131
Design Overview 27-131
Restart/Recovery 27-131
Security Considerations 27-132
I/O Specification 27-133
Output File 27-133
Design Assumptions 27-139
saexpuar (Export to Universal Account Reconciliation System from Sales Audit) 27-139
Schedule 27-139
Design Overview 27-139
Restart/Recovery 27-139
I/O Specification 27-140
Output File Layout 27-140
Design Assumptions 27-140
saprepost (Pre/Post Helper Processes for ReSA Batch Programs) 27-140
Schedule 27-141
Design Overview 27-141
Restart/Recovery 27-141
Design Assumptions 27-141

sapurge (Purge Aged Store/Day Transaction, Total Value and Error Data from Sales Audit) 27-141
Schedule 27-142
Design Overview 27-142
Restart/Recovery 27-143
Design Assumptions 27-143
b8d_sa_purge (Purge Into History Tables) 27-143
Design Overview 27-143
Scheduling Constraints 27-143
Restart/Recovery 27-144
Key Tables Affected 27-144

Send Us Your Comments

Oracle Retail Merchandising Foundation Cloud Service Operations Guide, Volume 1 - Batch Overviews and Designs, Release 19.0

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at http://www.oracle.com.

Preface

This *Oracle Retail Merchandising Foundation Cloud Service Operations Guide - Volume 1– Batch Overviews and Designs* provides critical information about the processing and operating details of the Oracle Retail Merchandising System (RMS), including the following:

- System configuration settings
- Technical architecture
- Functional integration dataflow across the enterprise
- Batch processing

Audience

This guide is for:

- Systems administration and operations personnel
- Systems analysts
- Business analysts who need information about Merchandising System processes and interfaces

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Integration Bus documentation set:

- *Oracle Retail Merchandising Foundation Cloud Service Release Notes*

- *Oracle Retail Merchandising Foundation Cloud Service Operations Guide, Volume 2 - Message Publication and Subscription Designs*
- *Oracle Retail Merchandising Foundation Cloud Service Administration Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Implementation Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Deals and Cost Changes User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Do the Basics Changes User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Finance User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Foundation Data User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Franchise User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Inventory User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Items User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Pricing User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Purchase Orders and Contracts User Guide*
- *Oracle Retail Merchandising Foundation Cloud Service Replenishment User Guide*

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following Web site:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. You can obtain them through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introduction

Welcome to the Oracle Retail Merchandising Operations Guide. The guide is designed to inform you about the 'backend' of Merchandising: data inputs, processes, and outputs. As a member of the Oracle Retail family, Merchandising provides the many benefits of enterprise application integration (EAI).

A primary benefit of EAI is the near real-time view of data that results from message-based processes between Merchandising and other products on the Oracle Retail Integration Bus (RIB). RIB integration allows Merchandising to overcome time lags to data updates. As a result, Merchandising is less dependent upon the batch window.

Contents of This Guide

The major components of the Operations Guide include the two volumes described below.

Volume 1 - Batch Overviews and Designs

Batch overviews tie a functional area description to the batch processes illustrated in the designs.

Batch designs describe how, on a technical level, an individual batch module works and the database tables that it affects. In addition, batch designs contain file layout information that is associated with the batch process.

Batch designs can be referenced by name through the table of contents of this volume.

Volume 2 - Message Publication and Subscription Designs

Oracle Retail Integration Bus (RIB) Merchandising functional overviews are incorporated into the publication and subscription designs. Therefore, the retailer can extract the business rationale behind each publication or subscription as well as the technical details that describe, on a technical level, how Merchandising publishes messages to the RIB or how Merchandising subscribes to messages from the RIB. A chapter in this volume also addresses how Merchandising utilizes the Oracle Retail Service Layer (RSL).

A Note about 'External' Subscription RIB APIs

Subscription APIs that are designated as 'External' are designed to be interfaces for external systems that maintain the applicable data. In other words, Merchandising is not the 'system of record' for maintaining the data. Instead, Merchandising subscribes

to consume the data when it is published so that the corresponding data in Merchandising can be kept in sync with the external system that maintains the data.

Volume 3 - Back-End Configuration and Operations

This volume describes the important features that necessary to run the Pro*C programs and the RETL programs associated with Merchandising. Additional Merchandising configuration and operations information is also included in this volume. Topics include:

- Pro*C Restart and Recovery
- Pro*C Multi-Threading
- Pro*C Array Processing
- Pro*C Input and Output Formats
- Internationalization
- Custom Post Processing
- Integrating RMS with Oracle Retail Workspace
- Setting up Oracle Business Intelligence Publisher

Merchandising Modules

For Merchandising retailers who purchase additional modules, the guide includes descriptions of the batch programs related to the following:

- Sales Audit
- Trade Management

Batch Schedule

The batch schedule is a program list with pre/post dependencies for each batch job. For each individual user, the schedule is a suggested starting point for the installation. Some programs are specific to products that may not be installed, so these programs may not be used at all.

Pro *C Input and Output Formats

Oracle Retail batch processing utilizes input from both tables and flat files. Further, the outcome of processing can both modify data structures and write output data. Interfacing Oracle Retail with external systems is the main use of file based I/O.

General Interface Discussion

To simplify the interface requirements, Oracle Retail requires that all in-bound and out-bound file-based transactions adhere to standard file layouts. There are two types of file layouts, detail-only and master-detail, which are described in the sections below.

An interfacing API exists within Oracle Retail to simplify the coding and the maintenance of input files. The API provides functionality to read input from files, ensure file layout integrity, and write and maintain files for rejected transactions.

Standard File Layouts

The Merchandising interface library supports two standard file layouts; one for master/detail processing, and one for processing detail records only. True sub-details are not supported within the Merchandising base package interface library functions.

A 5-character identification code or record type identifies all records within an I/O file, regardless of file type. The following includes common record type values:

- FHEAD—File Header
- FDETL—File Detail
- FTAIL—File Tail
- THEAD—Transaction Header
- TDETL—Transaction Detail
- TTAIL—Transaction Tail

Each line of the file must begin with the record type code followed by a 10-character record ID.

Detail-Only Files

File layouts have a standard file header record, a detail record for each transaction to be processed, and a file trailer record. Valid record types are FHEAD, FDETL, and FTAIL.

Example:

```
FHEAD0000000001STKU1996010100000019960929
FDETL0000000002SKU100000040000011011
FDETL0000000003SKU100000050003002001
FDETL0000000004SKU100000050003002001
FTAIL00000000050000000003
```

Master and Detail Files

File layouts consists of:

- Standard file header record
- Set of records for each transaction to be processed
- File trailer record.

The transaction set consists of:

- Transaction set header record
- Transaction set detail for detail within the transaction
- Transaction trailer record

Valid record types are FHEAD, THEAD, TDETL, TTAIL, and FTAIL.

Example:

```
FHEAD0000000001RTV 19960908172000
THEAD000000000200000000000001199609091202000000000003R
TDETL000000000300000000000001000001SKU10000012
TTAIL0000000004000001
THEAD000000000500000000000002199609091202001215720131R
TDETL000000000600000000000002000001UPC400100002667
TDETL000000000700000000000020000021UPC400100002643 0
```

```
TTAIL0000000008000002
FTAIL00000000090000000007
```

Table 1–1 File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type.
	File Line Identifier	Number(10	Specified by external system	Line number of the current file.
	File Type Definition	Char(4)	n/a	Identifies transaction type.
	File Create Date	Date	Create date	Date file was written by external system.
Transaction Header	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type.
	File Line Identifier	Number(10	Specified by external system	Line number of the current file.
	Transaction Set Control Number	Char(14)	Specified by external system	Used to force unique transaction check.
	Transaction Date	Char(14)	Specified by external system	Date the transaction was created in external system.
Transaction Detail	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type.
	File Line Identifier	Number(10	Specified by external system	Line number of the current file.
	Transaction Set Control Number	Char(14)	Specified by external system	Used to force unique transaction check.
	Detail Sequence Number	Char(6)	Specified by external system	Sequential number assigned to detail records within a transaction.
Transaction Trailer	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type.
	File Line Identifier	Number(10	Specified by external system	Line number of the current file.
	Transaction Detail Line Count	Number(6)	Sum of detail lines	Sum of the detail lines within a transaction.

Table 1–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type.
	File Line Identifier	Number(10	Specified by external system	Line number of the current file.
	Total Transaction Line Count	Number(10	Sum of all transaction lines	All lines in file less the file header and trailer records.

2

Administration Batch

This chapter contains information about a number of batch processes perform administrative processes in Merchandising. These processes range from incrementing the 'current business date for transactions' (known in Merchandising as vdate) to purging unused data and auditing database transactions.

Program Summary

Table 2–1 Program Summary

Program	Description
async_job_status_retry_ cleanup.ksh	Purge Asynchronous Job Tables
pre/post	Pre/Post Helper Processes for Batch Programs
dlyprg.pc	Daily Purge of Foundation Data
daily_purge_job	Daily Purge of Foundation Data
taxevntprg.pc	Tax Event Purge
tax_event_purge_job	Tax Event Purge
dtesys.pc	Increment Virtual Business Date
trunctbl	Truncate Table Script
rms_oi_purge.ksh	Purge Dashboard Working Tables
raf_notification_purge.ksh	Purge RAF Notifications
batch_archive_purge_ hist.ksh	Archive and Truncate Purge History Tables
admin_api_purge.ksh	Purge Manage Admin Records
refreshmview.ksh	Job to Refresh Materialized View
data_export_purge_job	Purging of All the Extracted Data
job_audit_logs_purge_job	Purge Old Job Auditing Logs

async_job_status_retry_cleanup.ksh (Purge Asynchronous Job Tables)

Module Name async_job_status_retry_cleanup.ksh

Description Purge Asynchronous Job Tables

Functional Area	Administration
Module Type	Admin
Module Technology	ksh
Catalog ID	RMS180
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job cleans up the Merchandising asynchronous jobs tables. The asynchronous job management tables (RMS_ASYNC_STATUS and RMS_ASYNC_RETRY) track each asynchronous call that is made. These tables are used to see error information and help with retrying failed calls.

This program will be run ad hoc and will accept a parameter of # days of information that will be deleted.

Restart/Recovery

N/A

Key Tables Affected

Table 2–2 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_ASYNC_STATUS	No	No	No	Yes
RMS_ASYNC_RETRY	No	No	No	Yes

Input/Out Specification

N/A

prepost (Pre/Post Helper Processes for Batch Programs)

Module Name	prepost.pc
Description	Pre/Post Helper Processes for Batch Programs
Functional Area	Administration
Module Type	Business Processing
Module Technology	ProC
Catalog ID	N/A Individual pre/post jobs have Catalog IDs
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The pre/post module facilitates multi-threading by allowing general system administration functions (such as table deletions or mass updates) to be completed after all threads of a particular program have been processed.

This program will take three parameters: username/password to log on to Oracle, a program before or after which this script must run and an indicator telling whether the script is a pre or post function. It will act as a shell script for running all pre-program and post-program updates and purges (the logic was removed from the programs themselves to enable multi-threading and restart/recovery).

Pre/Post contains the following helper functions, which are should be individually scheduled with the related main programs.

Table 2–3 Pre/Post Helper Functions

Catalog ID	Prepost Job	Related Main Program Catalog ID	Related Main Program
RMS400	prepost rpl pre	RMS315	rplext
RMS401	prepost salweek post	RMS346	salweek
RMS402	prepost salmth post	RMS343	salmth
RMS403	prepost rplapprv pre	RMS300	rplapprv
RMS404	prepost rplatupd pre	RMS313	rplatupd
RMS405	prepost rplatupd post	RMS313	rplatupd
RMS406	prepost rilmaint pre	RMS311	rilmaint
RMS407	prepost rilmaint post	RMS311	rilmaint
RMS408	prepost supmth post	RMS369	supmth
RMS409	prepost sccext post	RMS355	sccext
RMS410	prepost hstbld pre	RMS239	hstbld
RMS411	prepost hstbld post	RMS239	hstbld
RMS413	prepost edidlprd post	RMS47	edidlprd
RMS414	prepost edidlprd pre	RMS47	edidlprd
RMS417	prepost cntrordb post	RMS232	cntrordb
RMS418	prepost fsadnld post	N/A	N/A
RMS419	prepost btchcycl	N/A	No related main process. Is used to enable DB policies that might have been disabled in order to run batch.
RMS421	prepost poscdnld post	N/A	poscdnld
RMS423	prepost htsupld pre	N/A	htsupld
RMS425	prepost reclsdly pre	RMS302	reclsdly
RMS426	prepost reclsdly post	RMS302	reclsdly

Table 2–3 (Cont.) Pre/Post Helper Functions

Catalog ID	Prepost Job	Related Main Program Catalog ID	Related Main Program
RMS427	prepost ibcalc pre	RMS249	ibcalc
RMS428	prepost fcstprg pre	RMS227	fcstprg
RMS429	prepost fcstprg post	RMS249	fcstprg
RMS430	prepost reqext pre	RMS310	reqext
RMS431	prepost reqext post	RMS310	reqext
RMS432	prepost stkupd pre	N/A	Stkupd
RMS433	prepost replroq pre	RMS308	Replroq
RMS434	prepost rplext post	RMS315	Rplext
RMS438	prepost saleoh pre	RMS337	Saleoh
RMS440	prepost salweek pre	RMS346	salweek
RMS441	prepost dealinc pre	RMS211	Dealinc
RMS442	prepost dealday pre	RMS208	dealday
RMS443	prepost dealday post	RMS208	dealday
RMS444	prepost dealact_nor pre	RMS206	Dealact
RMS445	prepost dealact_po pre	RMS206	Dealact
RMS446	prepost dealact_sales pre	RMS206	Dealact
RMS447	prepost dealfct pre	RMS209	Dealfct
RMS448	prepost dealcls post	RMS209	Dealcls
RMS449	prepost hstbldmth post	RMS241	hstbldmth
RMS450	prepost vendinvc pre	N/A	vendinvc
RMS451	prepost vendinvf pre	N/A	vendinvf
RMS452	prepost vendinvc post	N/A	vendinvc
RMS453	prepost vendinvf post	N/A	vendinvf
RMS454	prepost docclose pre	RMS219	docclose
RMS455	prepost stkprg post	RMS360	stkprg
RMS456	prepost wfordupld pre	RMS392	wfordupld
RMS457	prepost wfretupld pre	N/A	wfretupld
RMS458	prepost replsizeprofile pre	RMS309	replsizeprofile
RMS459	prepost supsplit pre	RMS370	supsplit
RMS461	prepost batch_ ordcostcompupd pre	RMS190	batch_ ordcostcompupd
RMS462	prepost batch_ ordcostcompupd post	RMS190	batch_ ordcostcompupd

Table 2–3 (Cont.) Pre/Post Helper Functions

Catalog ID	Prepost Job	Related Main Program Catalog ID	Related Main Program
RMS463	prepost batch_ costcompupd post	RMS190	batch_ ordcostcompupd
RMS465	prepost dlyprg post	RMS218	dlyprg
RMS466	prepost tsfprg pre	RMS380	tsfprg
RMS467	prepost tsfprg post	RMS380	tsfprg
RMS468	prepost fcexec pre	RMS223	fcexec
RMS469	prepost start_batch pre	N/A	Sets the batch running ind to 'Y' to limit front end use of the system.
RMS470	prepost end_batch post	N/A	Sets the batch running ind to 'N' to reenable all front end use of the system. This should be the last job in the batch cycle.
RMS488	prepost btchcycl post	N/A	This job reenables all policies in the Merchandising owning schema.
RMS489	prepost dealfct post	RMS209	dealfct

Restart/Recovery

N/A

dlyprg (Daily Purge of Foundation Data)

Module Name	dlyprg.pc
Description	Daily Purge of Foundation Data
Functional Areas	Administration
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS218
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this program is to delete specific Foundation Data entities from Merchandising.

When users 'delete' a record in the Merchandising user interface, information is generally not immediately deleted at the database level; instead, data is marked as being in deleted status and also inserted into the DAILY_PURGE table.

Complex referential integrity relationships determine whether data can actually be deleted from the database (for example, a store cannot be deleted if any transactions related to the store are still on current transaction tables). Dlyprg.pc checks these complex rules. If the deletion request passes the rules, dlyprg.pc deletes the data. If dlyprg.pc is not able to delete the data, it writes a record to a log table for further investigation. Dlyprg will continue to attempt to delete marked data until all references have been purged from the system and the deletion of the foundation data entity finally succeeds.

Restart Recovery

This program has inherent restart ability. Records that have been successfully purged are deleted from the DAILY_PURGE table. This ensures that if the program is restarted, it does not attempt to delete records that have been previously processed.

I/O Specification

N/A

Design Assumptions

N/A

daily_purge_job (Daily Purge of Foundation Data)

Module Name	daily_purge_job
Description	Daily Purge of Foundation Data
Functional Areas	Administration
Module Type	Admin - Adhoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

When users 'delete' a record in the RMS user interface, information is generally not immediately deleted at the database level; instead, data is marked as being in deleted status and also inserted into the DAILY_PURGE table.

Thread assignment program (DAILY_PURGE_THREAD) will filter eligible records from daily purge (DAILY_PURGE) table wherein all entities ready for purging aside/except from Item-Location (ITEM_LOC table) records. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_DAILY_PURGE_STG.

The Business logic program (DAILY_PURGE) will process all records from the staging table. Using bulk processing, this program will delete specific Foundation Data

entities from RMS respective tables. Complex referential integrity relationships determine whether data can actually be deleted from the database (for example, a store cannot be deleted if any transactions related to the store are still on current transaction tables). This program checks these complex rules. If the deletion request passes the rules, this job will continues to delete the data. If it is not able to delete the data, it writes a record to the DAILY_PURGE_ERROR_LOG table for further investigation. This program will continue to attempt to delete marked data until all references have been purged from the system and the deletion of the foundation data entity finally succeeds. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 2–4 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart Recovery

NA

Key Tables Affected

Table 2–5 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_DAILY_PURGE_STG	Yes	No	No	No
DAILY_PURGE	Yes	No	No	Yes
DAILY_PURGE_ERROR_LOG	Yes	Yes	No	Yes
LOC_LIST_DETAIL	No	No	No	Yes
MONTH_DATA_BUDGET	Yes	No	No	Yes
HALF_DATA_BUDGET	Yes	No	No	Yes
VAT_DEPS	Yes	No	No	Yes
SKULIST_CRITERIA	Yes	No	No	Yes
DOMAIN_DEPT	Yes	No	No	Yes

Table 2–5 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
FORECAST_REBUILD	Yes	No	No	Yes
SUP_DATA	Yes	No	No	Yes
DEPT_SALES_HIST	Yes	No	No	Yes
DEPT_SALES_FORECAST	Yes	No	No	Yes
DEAL_ITEMLOC	Yes	No	No	Yes
DEPS	Yes	No	No	Yes
STOCK_LEDGER_INSERTS	Yes	No	No	Yes
STAKE_SCHEDULE	Yes	No	No	Yes
DEPT_CHRG_DETAIL	Yes	No	No	Yes
WH_DEPT	Yes	No	No	Yes
DEPT_CHRG_HEAD	Yes	No	No	Yes
SUP_BRACKET_COST	Yes	No	No	Yes
SUP_REPL_DAY	Yes	No	No	Yes
SUP_INV_MGMT	Yes	No	No	Yes
FILTER_GROUP_MERCH	Yes	No	No	Yes
IB_RESULTS	Yes	No	No	Yes
WEEK_DATA	Yes	No	No	Yes
DAILY_DATA	Yes	No	No	Yes
MONTH_DATA	Yes	No	No	Yes
TRAN_DATA_HISTORY	Yes	No	No	Yes
HALF_DATA	Yes	No	No	Yes
PARTNER	Yes	No	No	Yes
SHIPMENT	Yes	No	No	Yes
COST_ZONE_GROUP_ LOC	Yes	No	No	Yes
COST_ZONE	Yes	No	No	Yes
COST_ZONE_GROUP	Yes	No	No	Yes
UDA_ITEM_DEFAULTS	Yes	No	No	Yes
DOMAIN_CLASS	Yes	No	No	Yes
CLASS_SALES_HIST	Yes	No	No	Yes
CLASS_SALES_FORECAST	Yes	No	No	Yes
CLASS	Yes	No	No	Yes
DOMAIN_SUBCLASS	Yes	No	No	Yes
OTB	Yes	No	No	Yes
DIFF_RATIO_DETAIL	Yes	No	No	Yes
DIFF_RATIO_HEAD	Yes	No	No	Yes
SUBCLASS_SALES_HIST	Yes	No	No	Yes

Table 2–5 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SUBCLASS_SALES_FORECAST	Yes	No	No	Yes
SUBCLASS	Yes	No	No	Yes
MERCH_HIER_DEFAULT	Yes	No	No	Yes
WH	Yes	No	No	Yes
WH_ADD	Yes	No	No	Yes
STORE_SHIP_DATE	Yes	No	No	Yes
LOC_TRAITS_MATRIX	Yes	No	No	Yes
COST_ZONE_GROUP_LOC	Yes	No	No	Yes
ITEM_EXP_DETAIL	Yes	No	No	Yes
ITEM_EXP_HEAD	Yes	No	No	Yes
EXP_PROF_DETAIL	Yes	No	No	Yes
EXP_PROF_HEAD	Yes	No	No	Yes
STORE_GRADE_STORE	Yes	No	No	Yes
DAILY_SALES_DISCOUNT	Yes	No	No	Yes
LOAD_ERR	Yes	No	No	Yes
STORE	Yes	No	No	Yes
EDI_SALES_DAILY	Yes	No	No	Yes
COMP_STORE_LINK	Yes	No	No	Yes
SEC_GROUP_LOC_MATRIX	Yes	No	No	Yes
LOC_CLSF_HEAD	Yes	No	No	Yes
LOC_CLSF_DETAIL	Yes	No	No	Yes
SOURCE_DLVRY_SCHED	Yes	No	No	Yes
SOURCE_DLVRY_SCHED_DAYS	Yes	No	No	Yes
SOURCE_DLVRY_SCHED_EXC	Yes	No	No	Yes
COMPANY_CLOSED_EXCEP	Yes	No	No	Yes
LOCATION_CLOSED	Yes	No	No	Yes
POS_STORE	Yes	No	No	Yes
STORE_HIERARCHY	Yes	No	No	Yes
ADDR	Yes	No	No	Yes
TIF_EXPLODE	Yes	No	No	Yes
WALK_THROUGH_STORE	Yes	No	No	Yes
SKULIST_DETAIL	Yes	No	No	Yes
INV_STATUS_QTY	Yes	No	No	Yes

Table 2–5 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_ATTR_UPDATE_LOC	Yes	No	No	Yes
REPL_ATTR_UPDATE_HEAD	Yes	No	No	Yes
REPL_ATTR_UPDATE_ITEM	Yes	No	No	Yes
COST_SUSP_SUP_DETAIL	Yes	No	No	Yes
ITEM_HTS_ASSESS	Yes	No	No	Yes
ITEM_HTS	Yes	No	No	Yes
REQ_DOC	Yes	No	No	Yes
ITEM_IMPORT_ATTR	Yes	No	No	Yes
TIMELINE	Yes	No	No	Yes
COND_TARIFF_TREATMENT	Yes	No	No	Yes
ITEM_IMAGE	Yes	No	No	Yes
ITEM_SUPP_UOM	Yes	No	No	Yes
DEAL_SKU_TEMP	Yes	No	No	Yes
DEAL_DETAIL	Yes	No	No	Yes
ITEM_SUPP_COUNTRY	Yes	No	No	Yes
ITEM_SUPP_COUNTRY_DIM	Yes	No	No	Yes
RECLASS_ITEM	Yes	No	No	Yes
SUP_AVAIL	Yes	No	No	Yes
ITEM_SUPPLIER	Yes	No	No	Yes
ITEM_MASTER	Yes	No	No	Yes
PACK_TMPL_DETAIL	Yes	No	No	Yes
SUPS_PACK_TMPL_DESC	Yes	No	No	Yes
PACK_TMPL_HEAD	Yes	No	No	Yes
UDA_ITEM_LOV	Yes	No	No	Yes
UDA_ITEM_DATE	Yes	No	No	Yes
UDA_ITEM_FF	Yes	No	No	Yes
ITEM_SEASONS	Yes	No	No	Yes
ITEM_TICKET	Yes	No	No	Yes
COMP_SHOP_LIST	Yes	No	Yes	Yes
TICKET_REQUEST	Yes	No	No	Yes
PRICE_HIST	Yes	Yes	No	Yes
PACKITEM_BREAKOUT	Yes	No	No	Yes
PACKITEM	Yes	No	No	Yes
POS_MERCH_CRITERIA	Yes	No	No	Yes

Table 2–5 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_CHRG_HEAD	Yes	No	No	Yes
ITEM_CHRG_DETAIL	Yes	No	No	Yes
RECLASS_COST_CHG_ QUEUE	Yes	No	No	Yes
ITEM_PUB_INFO	Yes	No	No	Yes
ITEM_MFQUEUE	Yes	No	No	Yes
ITEM_XFORM_HEAD	Yes	No	No	Yes
ITEM_XFORM_DETAIL	Yes	No	No	Yes
DEAL_ITEM_LOC_ EXPLODE	Yes	No	No	Yes
ITEM_APPROVAL_ERROR	Yes	No	No	Yes

I/O Specification

NA

taxevntprg (Tax Event Purge)

Module Name	Taxevntprg
Description	Tax Event Purge
Functional Area	Purchase Order
Module Type	Admin
Module Technology	PROC
Catalog ID	RMS373
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch purges the tax events from the tax calculation event table. The records to be purged are based on its last update timestamp along with the tax event result.

Restart/Recovery

N/A

Key Tables Affected

Table 2–6 Key Tables Affected

Table	Select	Insert	Update	Delete
TAX_CALC_EVENT	No	No	No	Yes
PERIOD	Yes	No	No	Yes

Design Assumptions

N/A

tax_event_purge_job (Tax Event Purge)

Module Name tax_event_purge_job

Description Tax Event Purge

Functional Area Purchase Order

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (TAX_EVENT_PURGE_THREAD) will filter eligible records from tax calculated event (TAX_CALC_EVENT) table based on its purge criteria (retention number of days) with default value of 90 days and its tax event result defined as "C"ompleted Successfully. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_TAX_EVENT_PURGE_STG.

The Business logic program (TAX_EVENT_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from tax calculated event (TAX_CALC_EVENT) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 2–7 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 1 hour interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA

Table 2–7 (Cont.) Scheduling Constraints

Schedule Information	Description
Threading Scheme	NA

Restart/Recovery

NA

Key Tables Affected

Table 2–8 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_TAX_EVENT_PURGE_STG	Yes	Yes	No	Yes
TAX_CALC_EVENT	No	No	No	Yes
PERIOD	Yes	No	No	No

Input/Output Specification

NA

dtesys (Increment Virtual Business Date)

Module Name	dtesys.pc
Description	Increment Virtual Business Date
Functional Area	Administration
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS220
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program updates the PERIOD table for various dates required in Merchandising such as vdate, end-of-month and end-of-week dates.

Vdate (short for virtual business date) is used by Merchandising to maintain a consistent 'virtual' business date (without regard for actual date changes at midnight

or different dates in different timezone) for accounting purposes. Sysdate from the database is used to capture audit time and date stamps on transactions.

Note: Vdate is used to determine the business date for the financial impact of transactions.

Generally, dtesys is run without additional input parameters and increments the data by one day. However, if a specific date is passed into the program as a parameter, the system date will be updated to that date.

Special processing also occurs:

- Weekly

  When vdate = next_eow_date_unit, the program increments the last_eow_date_unit and next_eow_date_unit columns on system_variables. The last_eow_date_unit is updated to the current next_eow_date_unit and the next_eow_date_unit is updated to the next end-of-week date (calculated).

- Monthly

  When vdate = next_eom_date_unit, the program updates the last_eom_date_unit and next_eom_date_unit columns on system_variables. The last_eom_date_unit is updated to the current next_eom_date_unit and the next_eom_date_unit is updated to the next end-of-month date (calculated).

Restart/Recovery

N/A

I/O Specification

N/A

Design Assumptions

N/A

trunctbl.ksh (Truncate Table Script)

Module Name	trunctbl.ksh
Description	Truncate Table Script
Functional Area	Foundation
Module Type	Admin
Module Technology	KSH
Catalog ID	RMS475
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program performs truncate operations on a Merchandising table or a specific partition. It accepts an input table name and an optional partition name. If no partition name is passed, then the truncate is applied on the entire table.

This program must be run as either the Merchandising schema owner, or be run by a user that has been granted the following system privileges:

- drop any table
- alter any table

Currently, the following action and tables are processed by the batch. For the runtime parameters, refer to the *Oracle Retail Merchandising Batch Schedule*.

Table 2–9 Actions and Tables Processed by Batch

Table	Partition
NIL_INPUT_WORKING	N/A

Restart/Recovery

N/A

Key Tables Affected

N/A

Design Assumptions

N/A

rms_oi_purge.ksh (Purge Dashboard Working Tables)

Module Name	rms_oi_purge.ksh
Description	Purge data from the dashboard working tables
Functional Area	Operational Insight Dashboard Reports
Module Type	Admin
Module Technology	KSH
Catalog ID	RMS490
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program calls a package function to truncate the data in the Merchandising Operational Insight Dashboard staging tables. During normal operation, the staged data for the session are deleted when a user closes the report window. This program

provides a way to clean up and control the size of the staging tables if data failed to be deleted due to abnormal termination of the session.

Restart/Recovery

N/A

Key Tables Affected

Table 2–10 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_OI_BUYER_EARLY_LATE_SHIP	No	No	No	Yes
RMS_OI_BUYER_ORDERS_TO_APPROVE	No	No	No	Yes
RMS_OI_INV_ANA_OPEN_ORDER	No	No	No	Yes
RMS_OI_INV_ANA_VARIANCE	No	No	No	Yes
RMS_OI_INV_CTL_NEG_INV	No	No	No	Yes
RMS_OI_INV_ORD_ERRORS	No	No	No	Yes
RMS_OI_INV_ORD_ITEM_ERRORS	No	No	No	Yes
RMS_OI_MISSING_STOCK_COUNT	No	No	No	Yes
RMS_OI_OVERDUE_SHIP_ALLOC	No	No	No	Yes
RMS_OI_OVERDUE_SHIP_TSF	No	No	No	Yes
RMS_OI_OVERDUE_SHIP_RTV	No	No	No	Yes
RMS_OI_STK_ORD_PEND_CLOSE	No	No	No	Yes
RMS_OI_STOCK_COUNT_VARIANCE	No	No	No	Yes
RMS_OI_TSF_PEND_APPROVE	No	No	No	Yes
RMS_OI_UNEXPECTED_INV	No	No	No	Yes
RMS_OI_DATA_STWRD_INCOMP_ITEMS	No	No	No	Yes

Design Assumptions

N/A

raf_notification_purge.ksh (Purge RAF Notifications)

Module Name	raf_notification_purge.ksh
Description	Purge notifications from the Retail Application Framework table
Functional Area	Notifications
Module Type	Admin
Module Technology	Ksh
Catalog ID	RMS80
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program calls RAF_NOTIFICATION_TASK_PKG.DEL_NOTIF_PAST_RETENTION to delete notifications that are generated by Merchandising and Sales Audit and have passed the preconfigured number of retention days. This program provides a way to clean up and control the size of the RAF notification tables.

Restart/Recovery

N/A

Design Assumptions

N/A

batch_archive_purge_hist.ksh (Archive and Truncate Purge History Tables)

Module Name	batch_archive_purge_hist.ksh
Description	Archive and Truncate Purge History Tables
Functional Area	Administration
Module Type	Admin
Module Technology	ksh
Catalog ID	RMS477
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this program is to archive and truncate purge history tables regularly in Merchandising.

When you 'delete' a record in the Merchandising user interface, information is generally not immediately deleted at the database level; instead, data is marked as being in deleted status and also inserted into the DAILY_PURGE table. Next the purge processes will delete the data from Merchandising transaction tables. Before deleting data from these tables, transaction data will be archived by inserting into purge history tables by the transaction purge processes.

The batch_archive_purge_hist.ksh will export the purge history table data as a dump file using Oracle Data Pump export utility (expdp) and move the dump file to SFTP site for customer pick up. And after successful export of the transaction data, purge history tables are truncated.

This script has the below functions:

1. check_archive_dates - checks for the archive last run date. Based on the last archive date and the archive frequency input parameter, decides whether to archive and truncate the purge history tables or not. This ensures that even though this batch job is scheduled to run daily, the actual archiving and purging of the purge history tables will only occur every X number of days based on the input parameter.
2. truncate_prg_hist_tables - Truncates purge history tables after successful export of the transaction data.
3. update_rms_archive_date - update the Merchandising archive date, after the successful archiving and truncation of purge history tables.

Restart/Recovery

This program does not contain restart/recovery logic.

I/O Specifications

N/A

Design Assumptions

N/A

admin_api_purge.ksh (Purge Manage Admin Records)

Module Name	admin_api_purge.ksh
Description	Purge Manage Admin records
Functional Area	Administration
Module Type	Admin
Module Technology	ksh
Catalog ID	N/A
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script purges data from tables used for uploading Foundation Data from spreadsheets based on the retention days specified in the system parameter- PROC_DATA_RETENTION_DAYS for both Merchandising and Sales Audit and will help in keeping the size of these tables controlled.

Restart/Recovery

N/A

I/O Specifications

N/A

refreshmview.ksh (Forecast Roll Up Refresh Views)

Module Name	refreshmview.ksh
Description	Refreshes dept_sales_forecast, class_sales_forecast and subclass_sales_forecast materialized views
Functional Area	Financials
Module Type	Adhoc
Module Technology	ksh
Catalog ID	N/A
Runtime Parameters	Database connection, Materialized View Name, Nested(Valid values: Y - True and N - False).

Design Overview

This is a batch job that will refresh the specified materialized view. The materialized views refreshed are dept_sales_forecast, class_sales_forecast and subclass_sales_forecast.

This program will be run Adhoc and will accept materialized view name as the parameter. Nested refresh of the materialized view can be controlled using the optional parameter. By default, the refresh is nested.

Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	As Needed
Scheduling Considerations	It is a Adhoc
Pre-Processing	N/A

Schedule Information	Description
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_FORECAST	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
STORE	Yes	No	No	No
SUBCLASS_SALES_FORECAST	Yes	No	No	No
CLASS_SALES_FORECAST	Yes	No	No	No
DEPT_SALES_FORECAST	Yes	No	No	No

I/O Specification

N/A

data_export_purge_job (Purging of All the Extracted Data)

Module Name data_export_purge_job

Description Purging of all the extracted records (week old) for Xstore.

Functional Area Foundation1

Module Type Admin - Ad hoc

Module Technology Background processing

Catalog ID NA

Design Overview

This background job is composed of one step processing only. It will retain the business logic processing from original KSH script algorithm.

The Business logic program (DATA_EXPORT_SQL.PURGE_STG) will removed all old/aged records from the following staging tables related to data exported information which are considered week old regardless if data is extracted or not.

Scheduling Constraints

Table 2–11 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Adhoc
Frequency	Daily - 1 hour interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	NA

Restart/Recovery

NA

Key Tables Affected

Table 2–12 Key Tables Affected

Table	Select	Insert	Update	Delete
MERCHHIER_EXPORT_STG	No	No	No	Yes
ORGHIER_EXPORT_STG	No	No	No	Yes
STORE_EXPORT_STG	No	No	No	Yes
DIFFS_EXPORT_STG	No	No	No	Yes
DIFFGRP_EXPORT_STG	No	No	No	Yes
ITEM_EXPORT_STG	No	No	No	Yes
VAT_EXPORT_STG	No	No	No	Yes
RELITEM_EXPORT_STG	No	No	No	Yes
DATA_EXPORT_HIST	No	No	No	Yes

Integration Contract

NA

Design Assumptions

NA

job_audit_logs_purge_job (Purge Old Job Auditing Logs)

Module Name job_audit_logs_purge_job

Description Purge Old Job Auditing Logs

Functional Area Administration

Module Type Admin - Ad hoc

Module Technology	Background processing
Catalog ID	NA

Design Overview

This background job is composed of one step processing only. This new program/job will use the newly created support program maintain for purging records where affected table is partitioned.

The Business logic program (JOB_AUDIT_LOGS_PURGE) will invoke a call to a new program specific for handling historical logging table JOB_AUDIT_LOGS that is considered a partitioned table. PARTITION_SQL.PURGE_INTERVAL_PARTITION is called passing the target table name "JOB_AUDIT_LOGS" and will execute the proper deletion/purging of records from target table by exercising table partitioning handling such as Dropping Interval Partition (same as truncate or delete from table). There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop job with a flip of this indicator.

The purge program considered the system parameter setting, Job Logging History Months (job_log_hist_months) to determine those records that are older than a predetermined number of months.

Scheduling Constraints

Table 2–13 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 1 hour interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	NA

Restart/Recovery

NA

Key Tables Affected

Table 2–14 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
ALL_TAB_PARTITIONS	Yes	No	No	No
ALL_PART_TABLES	Yes	No	No	No

Table 2–14 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
JOB_AUDIT_LOGS	No	Yes	No	Yes

Design Assumptions

NA

3

Foundation Data Maintenance

Foundation Data is basic information that is required for Merchandising to function properly. Most foundation data is managed through the Merchandising user interface or integrations (often RIB) from external systems. However, there are some batch processes that relate to Foundation Data. This chapter describes the batch processes that are used to maintain general foundation data.

Programs in this chapter can be divided into five basic categories:

- Updates to Cost Components that must be applied to other foundation data and transactions
  - batch_compeffupd.ksh
  - batch_alloctsfupd.ksh
  - batch_depchrgupd.ksh
  - batch_expprofupd.ksh
  - batch_itmcostcompupd.ksh
  - batch_ordcostcompupd.ksh
  - elcexcprg.ksh
  - elc_except_purge_job
- Rebuilds of detail information for lists/groups
  - dfrtbld.pc
  - lclrbld.pc
  - loc_list_rebuild_job
  - batch_rfmvcurrconv.ksh
  - refmvlocprimadd.ksh
- Application of pending changes
  - cremhierdly.pc
  - reclsdly.pc
- Rollup of detailed information
  - supmth.pc
- Foundation Data Purges
  - admin_api_purge.ksh
  - schedprg.pc

- activity_sched_purge_job
- prchstprg.pc
- price_hist_purge_job

Note: For more information on Foundation Data, see the Item Maintenance chapter.

Batch Design Summary

The following batch designs are included in this functional area:

- batch_compeffupd.ksh (Update ELC Components)
- batch_expprofupd.ksh (Apply Pending Rate Changes to Expense Profiles)
- batch_depchrgupd.ksh (Apply Pending to Up-Charge Cost Component Changes to Departments)
- batch_itmcostcompupd.ksh (Apply Pending Item Cost Component Updates)
- batch_alloctsfupd.ksh (Update Allocation and Transfer Based on Changes to Up-Charges)
- batch_ordcostcompupd.ksh (Apply Pending Cost Component and ELC Changes to Purchase Orders)
- elcexcprg.pc (Purge Aged Cost Component Exceptions)
- elc_except_purge_job (Purge Aged Cost Component Exceptions)
- dfrtbld.pc (Build Diff Ratios Based on Sales History)
- lclrbld.pc (Rebuild Dynamic Location Lists)
- loc_list_rebuild_job (Rebuild Dynamic Location Lists)
- batch_rfmvcurrconv.ksh (Refresh Currency Conversion Materialized View)
- refmvlocprimadd.ksh (Refresh Address Materialized View)
- cremhierdly.pc (Process Pending Merchandise Hierarchy Changes from External Systems)
- reclsdly.pc (Reclassify Items in Merchandise Hierarchy)
- supmth.pc (Rollup of Supplier Data)
- schedprg.pc (Purge Aged Store Ship Schedule)
- activity_sched_purge_job (Purge Aged Store Ship Schedule)
- prchstprg.pc (Purge Aged Price History Data)
- price_hist_purge_job (Purge Aged Price History Data)
- tcktdnld (Download of Data to be Printed on Tickets)
- refmvl10entity (Refresh MV MV_L10N_ENTITY)

admin_api_purge (Purge Manage Admin records)

Module Name admin_api_purge.ksh

Description	Purge Manage Admin records
Functional Area	Administration
Module Type	Admin
Module Technology	ksh
Catalog ID	
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script purges data from tables used for uploading Foundation Data from spreadsheets based on the retention days specified in the system parameter- PROC_DATA_RETENTION_DAYS for both Merchandising and Sales Audit and will help in keeping the size of these tables controlled.

Restart/Recovery

N/A

I/O Specification

N/A

batch_compeffupd (Update ELC Components)

Module Name	batch_compeffupd.ksh
Description	Apply Pending Cost Component, Up-charge and ELC Changes
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	KSH
Catalog ID	RMS185
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, users are allowed to make rate changes to cost components, up-charges and expense profiles and assign future effective dates to the changes. Additionally, when these future rate changes are specified, users can choose to cascade

these changes to lower levels. The options for how the updates can be cascaded are described in the table below:

Table 3–1 Options for Cascading Updates

Updated Entity	Cascade Options
Expense Profiles (Country, Supplier, or Partner)	Order, Item
Cost Component (Expense)	Country, Supplier, Partner, Item, Order
Cost Component (Assessment)	Item, Order
Cost Component (Up-charge)	Department, Item, Transfer/Allocation
Department Level Up-Charges	Item, Transfer/Allocation

This batch process is used to process updates to cost components of all types at the expense component level, updates to department level up-charges, and updates to expense profiles at the supplier, country, or partner level. The cascading to other levels is handled in the dependent processes which are run after this process:

- Allocation and Transfer Up-charge Update (batch_alloctsfupd)
- Expense Profile Update (batch_expprofupd)
- Item Cost Component Update (batch_itmcostcompupd)
- Purchase Order Cost Component Update (batch_ordcostcompupd)
- Department Up-charge (batch_depchrgupd)

Restart/Recovery

N/A

Design Assumptions

N/A

batch_expprofupd (Apply Pending Rate Changes to Expense Profiles)

Module Name	batch_expprofupd.ksh
Description	Apply Pending Rate Changes to Expense Profiles
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS188
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, you are allowed to make rate changes to expense type cost components and assign future effective dates to the changes. Additionally, when these future rate changes are specified, you can choose to cascade these changes to lower levels. For expense type cost components, this includes the ability to cascade the changes to country, supplier, and partner expense profiles. This script will process the updates to country, supplier, and partner expense profiles once the rate changes reach their effective date.

Restart/Recovery

N/A

Design Assumptions

N/A

batch_depchrgupd (Apply Pending Up-Charge Cost Component Changes to Departments)

Module Name	batch_depchrgupd.ksh
Description	Apply Pending Up-Charge Cost Component Changes to Departments
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS186
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, you are allowed to make rate changes to up-charges and assign future effective dates for the updates. Additionally, when these future rate changes are specified, you can choose to cascade these changes to lower levels. For up-charges, this includes the ability to cascade the changes made at the cost component level (for up-charge components) to department level up-charges. This script will process the updates to department level up-charges once the rate changes reach their effective date.

Restart/Recovery

N/A

Design Assumptions

N/A

batch_itmcostcompupd (Apply Pending Item Cost Component Updates)

Module Name	batch_itmcostcompupd.ksh
Description	Apply Pending Item Cost Component Updates
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS189
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, you are allowed to make rate changes to cost components, up-charges and expense profiles and assign future effective dates to the changes. Additionally, when these future rate changes are specified, you can choose to cascade these changes to lower levels. For items, changes can be cascaded down from each of the different types:

- Expense Profiles (country, supplier, or partner)
- Cost Components (expense, assessment, or up-charge)
- Department-level Up-charges

This script will process the updates for items for each of these types of rate updates once the rate changes reach their effective date.

Restart/Recovery

N/A

Design Assumptions

N/A

batch_alloctsfupd (Update Allocation and Transfer Based on Changes to Up-Charges)

Module Name	batch_alloctsfupd.ksh
Description	Update Allocation and Transfer Based on Changes to Up-Charges
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	ksh

Catalog ID	RMS184
Wrapper Script	wmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, you are allowed to make rate changes to up-charge cost components and department level up-charges and assign future effective dates to the changes. One of the things that can be designated when these future rate changes are specified is whether this update should also impact any open transfers or allocations with items in the department. If they have been flagged to update open transfers and allocations, then this script will process the updates once they reach their effective date.

Restart/Recovery

N/A

Design Assumptions

N/A

batch_ordcostcompupd (Apply Pending Cost Component and ELC Changes to Purchase Orders)

Module Name	batch_ordcostcompupd.ksh
Description	Apply Pending Cost Component and ELC Changes to Purchase Orders
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS190
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, you are allowed to make rate changes to cost components and expense profiles and assign future effective dates for the updates. Additionally, when these future rate changes are specified, you can choose to cascade these changes to lower levels. For orders, changes can be cascaded down from each of the different types:

- Expense Profiles (country, supplier, or partner)

- Cost Components (expense or assessment)

This script will process the updates for open orders for each of these types of rate updates once the rate changes reach their effective date.

Restart/Recovery

N/A

Design Assumptions

N/A

elcexcprg (Purge Aged Cost Component Exceptions)

Module Name	ELCEXCPRG.PC
Description	Purge Aged Cost Component Exceptions
Functional Area	Costing
Module Type	Admin
Module Technology	ProC
Catalog ID	RM S222
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, you are allowed to make rate changes to cost components, up-charges and expense profiles with future effective dates. Additionally, when these future rate changes are specified, you can choose to cascade these changes to lower levels. The options for how the updates can be cascaded are described in the table below:

Table 3–2 ELCEXCPRG.PC - Cascade Options

Updated Entity	Cascade Options
Expense Profiles (Country, Supplier, or Partner)	Order, Item
Cost Component (Expense)	Country, Supplier, Partner, Item, Order
Cost Component (Assessment)	Item, Order
Cost Component (Up-charge)	Department, Item, Transfer/Allocation
Department Level Up-Charges	Item, Transfer/Allocation

When the processes that apply these changes run, they may raise exceptions if the rate for an entity has been overwritten prior to the application of the future rate change. If

so, then exceptions are written to the COST_COMP_EXC_LOG table. This program purges the records from this table based on a number of retention months that is passed as a runtime parameter.

Restart/Recovery

N/A

Design Assumptions

N/A

elc_except_purge_job (Purge Aged Cost Component Exceptions)

Module Name	rtvprg.pc
Description	Purge Aged Returns to Vendors
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS320
Runtime Parameters	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

In RMS, users are allowed to make rate changes to cost components, up-charges and expense profiles with future effective dates. Additionally, when these future rate changes are specified, users can choose to cascade these changes to lower levels. The options for how the updates can be cascaded are described in the table below:

Updated Entity	Cascade Options
Expense Profiles (Country, Supplier, or Partner)	Order, Item
Cost Component (Expense)	Country, Supplier, Partner, Item, Order
Cost Component (Assessment)	Item, Order
Cost Component (Up-charge)	Department, Item, Transfer/Allocation
Department Level Up-Charges	Item, Transfer/Allocation

When the processes that apply these changes run, they may raise exceptions if the rate for an entity has been overwritten prior to the application of the future rate change. If so, then exceptions are written to the COST_COMP_EXC_LOG table.

Thread assignment program (ELC_EXC_PURGE_THREAD) will filter eligible records from cost component exceptions log (COST_COMP_EXC_LOG) table based on its purge criteria from defined number of retention months (default to 6 months). These

records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_ELC_EXC_PURGE_STG.

The Business logic program (ELC_EXC_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from cost component exceptions log (COST_COMP_EXC_LOG) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 3–3 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 3–4 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_ELC_EXC_PURGE_STG	Yes	Yes	No	Yes
COST_COMP_EXC_LOG	No	No	No	Yes

Design Assumptions

NA

dfrtbld (Build Diff Ratios Based on Sales History)

Module Name dfrtbld.pc

Description Build Diff Ratios Based on Sales History

Functional Area Foundation Data

Module Type Business Processing

Module Technology	ProC
Catalog ID	RM S214
Wrapper Script	rmswrap_multi_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Diff ratios are used by Merchandising as a way to assign a ratio to a group of diffs or diff combinations based on sales history. The parameters for how these are created are setup online in Merchandising and include specifying a subclass and one or more diff groups for a particular date range. Users also specify how often the ratios should be refreshed and what types of sales should be considered, regular, promotional and/or clearance.

For ratios that are due to be rebuilt, this batch program uses this information and summarizes the total sales for items with the subclass and diff groups selected. It then calculates a percent to each diff combination/store. Diff ratios are used for PO distribution within Merchandising.

Restart/Recovery

This program is for multithreading and restart/recovery.

Key Tables Affected

Table 3–5 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
DIFF_RATIO_HEAD	Yes	No	Yes	No
DIFF_RATIO_DETAIL	No	No	No	Yes
DIFF_GROUP_DETAIL	Yes	No	No	No
V_RESTART_DEPT	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC_HIST	Yes	No	No	No

I/O Specification

This batch will create a comma delimited output data file for sql loader to upload data to table DIFF_RATIO_DETAIL. The control script for the sql loader is dfrtbld.ctl.

Output File Layout

Table 3–6 dfrtbld.pc - Input File Layout

Field Name	Field Type	Default Value	Description
Diff_ratio_id	N/A	N/A	N/A

Table 3–6 (Cont.) dfrtbld.pc - Input File Layout

Field Name	Field Type	Default Value	Description
Seq_no	N/A	N/A	N/A
store	N/A	N/A	N/A
Diff_1	N/A	N/A	N/A
Diff_2	N/A	N/A	N/A
Diff_3	N/A	N/A	N/A
qty	N/A	N/A	N/A
pct	N/A	N/A	N/A

Design Assumptions

N/A

lclrbld (Rebuild Dynamic Location Lists)

Module Name	lclrbld.pc
Description	Rebuild Dynamic Location Lists
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS255
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is used to rebuild dynamic location lists based on the criteria defined when the location list was created. Once run, the location list will be updated to include only the locations that currently meet the defined criteria for the list, including adding any new locations. Any locations which no longer fit the criteria will be removed.

Restart/Recovery

The logical unit of work for this program is a location list. The restart location list view is used for threading. Table-based restart/recovery is used by the batch program.

Key Tables Affected

Table 3–7 Key Tables Affected

Table	Select	Insert	Update	Delete
LOC_LIST_HEAD	Yes	No	Yes	No

Table 3–7 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
LOC_LIST_DETAIL	Yes	Yes	No	Yes

Design Assumptions

N/A

loc_list_rebuild_job (Rebuild Dynamic Location Lists)

Module Name	loc_list_rebuild_job
Description	Rebuild Dynamic Location Lists
Functional Area	Foundation Data
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (LOC_LIST_REBUILD_THREAD) will filter eligible records from location list header (LOC_LIST_HEAD) table which are based on the criteria defined when it was created (LOC_LIST_CRITERIA table). These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_LOC_LIST_REBUILD_STG.

The Business logic program (LOC_LIST_REBUILD) will process all records from the staging table. Using bulk processing, this program will rebuild the location lists. Once run, the location list will be updated to include only the locations that currently meet the defined criteria for the list, including adding any new locations. Any locations which no longer fit the criteria will be removed. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 3–8 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad-Hoc
Frequency	Daily - 1hour interval - 12 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by location list

Restart/Recovery

NA

Key Tables Affected

Table 3–9 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_LOC_LIST_REBUILD_STG	Yes	Yes	No	Yes
LOC_LIST_HEAD	Yes	No	Yes	No
LOC_LIST_DETAIL	Yes	Yes	No	Yes

Design Assumptions

NA

batch_rfmvcurrconv (Refresh Currency Conversion Materialized View)

Module Name	batch_rfmvcurrconv.ksh
Description	Refresh Currency Conversion Materialized View
Functional Area	Foundation Data
Module Type	Admin
Module Technology	ksh
Catalog ID	RMS193
Wrapper Scripts	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script refreshes the materialized view MV_CURRENCY_CONVERSION_RATES.

Restart/Recovery

N/A

Key Tables Affected

Table 3–10 Key Tables Affected

Table	Select	Insert	Update	Delete
MV_CURRENCY_ CONVERSION_RATES	Yes	Yes	Yes	Yes
CURRENCY_RATES	Yes	No	No	No
EURO_EXCHANGE_RATE	Yes	No	No	No

Design Assumptions

N/A

refmvlocprimaddr (Refresh Address Materialized View)

Module Name refmvlocprimaddr.pc

Description Refresh Address Materialized View

Functional Area Foundation Data

Module Type Admin

Module Technology ProC

Catalog ID RMS305

Wrapper Script rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program refreshes the materialized view for location/primary address based on the address and warehouse tables. The view will contain primary address information for all locations, including company stores, customer stores, physical and virtual warehouses and external finishers.

Restart/Recovery

N/A

Key Tables Affected

Table 3–11 Key Tables Affected

Table	Select	Insert	Update	Delete
ADDR	Yes	No	No	No
WH	Yes	No	No	No

Design Assumptions

N/A

cremhierdly (Process Pending Merchandise Hierarchy Changes from External Systems)

Module Name	cremhierdly.pc
Description	Process Pending Merchandise Hierarchy Changes from External Systems
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS204
Runtime Parameters	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program reads merchandise hierarchy records from the pending merchandise hierarchy table whose effective date is tomorrow or earlier. The pending merchandise hierarchy table is populated by the Merchandise Hierarchy Reclass Subscription API. Each record is then used to either insert or update existing merchandise hierarchy data in Merchandising based on the action and hierarchy types. The inserted/updated records are deleted from the pending merchandise hierarchy table after they have been successfully processed.

This program is only required if updates to the merchandise hierarchy in Merchandising are being managed outside the application.

Restart/Recovery

This program is setup for multithreading and restart/recovery.

Key Tables Affected

Table 3–12 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
PEND_MERCH_HIER	Yes	No	No	Yes
PEND_MERCH_HIER_TL	No	No	No	Yes
DIVISION	No	Yes	Yes	No
GROUPS	No	Yes	Yes	No

Table 3–12 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
DEPS	No	Yes	Yes	No
CLASS	No	Yes	Yes	No
SUBCLASS	No	Yes	Yes	No

Design Assumptions

N/A

reclsdly (Reclassify Items in Merchandise Hierarchy

Module Name	Reclsdly.pc
Description	Reclassify Items in Merchandise Hierarchy
Functional Area	Foundation
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS302
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to reclassify items from one department/class/subclass combination to another. Reclassification events that are due to go into effect the next day are processed by this batch process. Before the reclassification is executed, validation is performed to make sure that there are no issues which would prevent the reclassification from moving forward. If not, then the updates are made to update the item's merchandise hierarchy, as well as other related updates, such as moving the value of the inventory in the stock ledger and notifying the Pricing service of the update. Any issues that prevent the item from being reclassified raise a non-fatal error in the program and write the error to the mass change rejections table.

Restart/Recovery

The logical unit of work is the combination of the reclass number and item. Restart ability is also based on reclass number and item.

Key Tables Affected

Table 3–13 Key Tables Affected

Table	Select	Insert	Update	Delete
RECLASS_ITEM	Yes	No	No	Yes
RECLASS_HEAD	Yes	No	No	Yes

Table 3–13 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
RECLASS_HEAD_TL	No	No	No	Yes
ITEM_MASTER	Yes	No	Yes	No
DEPS	Yes	No	No	No
GROUPS	Yes	No	No	No
PACKITEM	Yes	No	No	No
DEAL_ITEM_LOC_ EXPLODE	Yes	No	No	Yes
DEAL_ITEMLOC	Yes	No	No	No
DEAL_HEAD	Yes	No	No	No
ORDHEAD	Yes	No	Yes	No
ORDSKU	Yes	No	No	No
DEAL_CALC_QUEUE	Yes	Yes	No	No
HIST_REBUILD_MASK	No	Yes	No	No
RECLASS_ERROR_LOG	No	Yes	Yes	Yes
STAKE_SKU_LOC	Yes	Yes	Yes	Yes
ITEM_LOC_SOH	Yes	No	Yes	No
REPL_ITEM_LOC_ UPDATES	No	Yes	No	No
TRAN_DATA	No	Yes	No	No
SKULIST_DEPT	Yes	Yes	No	No
MC_REJECTIONS	No	Yes	No	No
RPM_ITEM_ MODIFICATION	No	Yes	Yes	No

Design Assumptions

N/A

supmth (Rollup of Supplier Data)

Module Name	supmth.pc
Description	Rollup of Supplier Data
Functional Area	Inventory
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS369
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The primary function of this batch is to convert daily transaction data to monthly data. After all data is converted, the daily information is deleted to reset the system for the next period. This is done by the batch's post processing function in prepost.

This module accumulates supplier data amounts by department/supplier/transaction type and creates or updates one supplier month row for each department/supplier combination. Based on the transaction type on supplier data, the following transactions are written to supplier month:

- type 1 – purchases at cost (written for consignment sales and orders received at POS or online)
- type 2 – purchases at retail (written for consignment sales and orders received at POS or online)
- type 3 – claims at cost (written for claim dollars refunded on RTV orders)
- type 10 – markdowns at retail (net amount based on markdowns, markups, markdown cancellations and markup cancellations)
- type 20 – order cancellation costs (written for all supplier order cancellations)
- type 30 – sales at retail (written for consignment stock sales)
- type 40 – quantity failed (written for QC shipments with failed quantities)
- type 70 – markdowns at cost (net amount based on supplier cost markdowns)

Restart/Recovery

The logical unit of work is dept, supplier.

Key Tables Affected

Table 3–14 Key Tables Affected

Table	Select	Insert	Update	Delete
SUP_DATA	Yes	No	No	No
SUP_MONTH	No	Yes	No	No
SYSTEM_VARIABLES	Yes	No	No	No

Design Assumptions

N/A

schedprg (Purge Aged Store Ship Schedule)

Module Name schedprg.pc

Description Purge Aged Store Ship Schedule

Functional Area Foundation Data

Module Type	Admin
Module Technology	ProC
Catalog ID	RMS356
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will purge all old records related to store ship dates and location and company closed dates and exceptions. Old records are determined by the Ship Schedule History months and Location Closed History months system parameters.

Restart/Recovery

This program will use the commit max counter on the restart control table to periodically commit delete operations. Periodic commits are performed to ensure that rollback segments are not exceeded in case of considerable volume.

Key Tables Affected

Table 3–15 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
STORE_SHIP_DATE	No	No	No	Yes
COMPANY_CLOSED_ EXCEP	No	No	No	Yes
COMPANY_CLOSED	No	No	No	Yes
COMPANY_CLOSED_TL	No	No	No	Yes
LOCATION_CLOSED	No	No	No	Yes
LOCATION_CLOSED_TL	No	No	No	Yes

Design Assumptions

N/A

activity_sched_purge_job (Purge Aged Store Ship Schedule)

Module Name	activity_sched_purge_job
Description	Purge Aged Store Ship Schedule
Functional Area	Foundation Data
Module Type	Admin - Ad hoc
Module Technology	Background Processing

Catalog ID

Runtime Parameters Database connection

Design Overview

This background job is composed of two steps of processing. It will have a threading assignment and a business logic processing.

Thread assignment program (SCHED_PURGE_THREAD) will filter eligible records from location closed (LOCATION_CLOSED and LOCATION_CLOSED_TL), company closed exceptions (COMPANY_CLOSED_EXCEP) and company closed (COMPANY_CLOSED and COMPANY_CLOSED_TL) tables based on its purge criteria from system parameter settings. The Location Closed History Months (loc_close_hist_months) parameter will determine how long a location and/or company with close date should remain on the associated tables. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_SCHED_PURGE_STG.

The Business logic program (SCHED_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from location closed (LOCATION_CLOSED and LOCATION_CLOSED_TL), company closed exceptions (COMPANY_CLOSED_EXCEP) and company closed (COMPANY_CLOSED and COMPANY_CLOSED_TL) tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 3–16 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 3–17 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No

Table 3–17 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_SCHED_PURGE_STG	Yes	Yes	No	Yes
COMPANY_CLOSED_ EXCEP	No	No	No	Yes
COMPANY_CLOSED	No	No	No	Yes
COMPANY_CLOSED_TL	No	No	No	Yes
LOCATION_CLOSED	No	No	No	Yes
LOCATION_CLOSED_TL	No	No	No	Yes

Design Assumptions

NA

prchstprg(Purge Aged Price History Data)

Module Name	prchstprg.pc
Description	Purge Aged Price History Data
Functional Area	Foundation Data
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS298
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program deletes price history records, which are older than the price history retention days system parameter.

This program ensures that the most recent price history record for the item/location/transaction type combination is preserved and deletes all aged records.

Restart/Recovery

This program will use the commit_max_ctr on the restart_control table to periodically commit SQL delete operations. Restart/Recovery is achieved by processing records that have not been deleted. The restart bookmark table stores the current partition position as the bookmark string to restart a thread.

However, in cases where the price history table is very large, a particularly large rollback segment may be specified to reduce the risk of exceeding rollback segment

space. This will depend on the size of normal rollback segments and the size of the price history table.

Performance Considerations

The commit max counter field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O. The recommended commit counter setting is 10000 records (subject to change based on experimentation). In case the price history table is very large then the number of partitions on the table may be increased and then after the number of threads for this program should be increased.

Key Tables Affected

Table 3–18 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
PRICE_HIST	No	No	No	Yes
DBA_TAB_PARTITIONS	Yes	No	No	No

Design Assumptions

N/A

price_hist_purge_job (Purge Aged Price History Data)

Module Name	price_hist_purge_job
Description	Purge Aged Price History Data
Functional Area	Foundation Data
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (PRICE_HIST_PURGE_THREAD) will filter eligible records from price history (PRICE_HIST) tables based on its purge criteria from system parameter settings which is Price History Retention Days (orice_hist_retention_days). These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_PRICE_HIST_PURGE_STG.

The Business logic program (PRICE_HIST_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the old/aged records from price history (PRICE_HIST) table and keep only the most recent records for the item/location/tran type combinations. It will free up and clean the staging table

afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 3–19 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 3–20 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_PRICE_HIST_PURGE_STG	Yes	Yes	No	Yes
PRICE_HIST	No	No	No	Yes
DBA_TAB_PARTITIONS	Yes	No	No	No

tcktdnld (Download of Data to be Printed on Tickets)

Module Name tcktdnld.pc

Description Download of Data to be Printed on Tickets

Functional Area Foundation Data

Module Type Integration

Module Technology PROC

Catalog ID RMS59

Wrapper Script rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program creates an output file containing the information to be printed on a ticket or label for a particular item/location. This program is driven by the requests for tickets generated from Merchandising and Pricing. The details of what should be printed on each ticket are defined in Merchandising on the ticket type details table.

Restart/Recovery

N/A

Key Tables Affected

Table 3–21 Key Tables Affected

Table	Select	Insert	Update	Delete
TICKET_REQUEST	Yes	No	No	Yes
STORE	Yes	No	No	No
TICKET_TYPE_HEAD	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
TICKET_TYPE_DETAIL	Yes	No	No	No
UDA_VALUES	Yes	No	No	No
UDA_VALUES_TL	Yes	No	No	No
UDA_ITEM_LOV	Yes	No	No	No
UDA	Yes	No	No	No
UDA_TL	Yes	No	No	No
UDA_ITEM_FF	Yes	No	No	No
UDA_ITEM_FF_TL	Yes	No	No	No
UDA_ITEM_DATE	Yes	No	No	No
ITEM_TICKET	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_SUPPLIER	Yes	No	No	No
ITEM_SUPP_COUNTRY_DIM	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
DEPS	Yes	No	No	No
DEPS_TL	Yes	No	No	No
CLASS	Yes	No	No	No
CLASS_TL	Yes	No	No	No
SUBCLASS	Yes	No	No	No

Table 3–21 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SUBCLASS_TL	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ORDSKU	Yes	No	No	No
WH	Yes	No	No	No
VAT_ITEM	Yes	No	No	No
RPM_PC_TICKET_REQUEST	Yes	No	No	Yes
GTAX_ITEM_ROLLUP	Yes	No	No	No

I/O Specification

Integration Type Download from Merchandising

File Name Determined by runtime parameters

Integration Contract IntCon000107

Output File Layout

Table 3–22 tcktdnld.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	File Type Definition	Char(4)	TCKT	Identifies file as 'Print Ticket Requests'
	File Create Date	Char(14)	N/A	The date on which the file was created in 'YYYMMDDHHMISS' format

Table 3–22 (Cont.) tcktdnld.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
THEAD	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	ITEM	Char(25)	N/A	ID number of the transaction level item for which the ticket applies.
	Ticket Type	Char(4)	N/A	ID which indicates the ticket type to be printed
	Location Type	Char(1)	N/A	Identifies the type of location for which tickets will be printed. Valid values are store (S) and warehouse (W).
	Location	Char(10)	N/A	The ID of the store or warehouse for which tickets will be printed
	Quantity	Number(12,4)	N/A	The quantity of tickets to be printed; which includes 4 implied decimal places
TCOMP	File Type Record Descriptor	Char(5)	TCOMP	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	ITEM	Char(25)	N/A	ID number of the item which is only populated if the item in THEAD is a pack item
	Quantity	Number(12,4)	N/A	Quantity of the component item as a part of the pack; includes 4 implied decimal places

Table 3–22 (Cont.) tcktdnld.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TDETL	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	Detail Sequence Number	Number(10)	N/A	Sequential number assigned to the detail records
	Ticket Item	Char(4)	N/A	ID indicating the detail to be printed on the ticket. If the attribute is a UDA, then this will contain the ID of the UDA. Otherwise, it is the code associated with the attribute in Merchandising (such as, CLSS = class)
	Attribute Description	Char(120)	N/A	Description of the attribute – either the UDA description or the Merchandising description for the attribute
	Value	Char(250)	N/A	Detail to be printed on the ticket (for example:. Item number, Department Number, Item description)
	Supplement	Char(120)	N/A	Supplemental description to the Value (for example: Department Name)
TTAIL	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	Transaction Detail Line Count	Number(6)	sum of detail lines	Sum of the detail lines within a transaction
FTAIL	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file

Design Assumptions

N/A

refmvl10entity (Refresh MV MV_L10N_ENTITY)

Module Name REFMVL10ENTITY.PC

Description Refresh Materialized view MV_L10N_ENTITY

Functional Area	Administration
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS304
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program refreshes the materialized view MV_L10N_ENTITY that is based on ADDR, OUTLOC, COMPHEAD, COUNTRY_ATTRIB table.

Restart/Recovery

This batch program uses table-based restart/recovery.

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

I/O Specification

N/A

likestorebatch (Like Store Batch Processing)

Module Name	likestorebatch.ksh
Description	Like Store Batch Processing
Functional Area	Foundation
Module Type	Business Processing
Module Technology	Ksh
Catalog ID	N/A
Runtime Parameters	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to process stores from the store add staging table with like stores to copy attributes and items from an existing store to a new store.

Previously, the like store functionality was also processed within the store add asynchronous process. However, this posed an issue when the like store process abnormally ends, which will hold up the store add process. There was also a performance consideration with the like store process, as it was possible that a single like store can have millions of items, which will take a long time to process, thus preventing the store add asynchronous process to process new records. The like store process has been decoupled from the store add program and now runs as a separate hourly batch job, removing the dependency between both processes.

The like store batch program picks up all rows from the store add staging table wherein the process status is set to 02STOREADD_POST and the like store column is populated. It will then gather all items associated to the like store and explode this to the like store staging table and process all the inserted records by chunk. Chunking is based on the system parameter maximum chunk size, and it should be noted that there is no sorting or grouping done when chunking the rows.

For each chunk, records are inserted into the temporary table for store add, which will serve as the driving table for the like store process of each thread.

For each successfully processed chunk, it will delete all the matching rows from the like store staging table. Once all rows are processed, the process status column is updated for the specific store, depending on whether there are records remaining in the like store staging table for that store. If there are no more entries for a store, then the store will be deleted from the store add table. If there are entries remaining, then the status will be updated to 05LIKESTORE_FAIL.

Restart/Recovery

In case of failure, the like store batch will not pick up any new entries from the store add table until the issue has been rectified. Errors are determined by looking up like store staging, if there are any rows left from the previous run. Successfully processed records are deleted from the staging table.

Key Tables Affected

Table 3–23 Key Tables Affected

Table	Select	Insert	Update	Delete
STORE_ADD	Yes	No	Yes	Yes
ITEM_EXP_HEAD	No	Yes	No	No
ITEM_EXP_DETAIL	No	Yes	No	No
ITEM_LOC	No	Yes	No	No
ITEM_LOC_SOH	No	Yes	No	No
PRICE_HIST	No	Yes	No	No
ITEM_SUPP_COUNTRY_LOC	No	Yes	No	No

Table 3–23 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_ITEM_LOC	No	Yes	No	No
REPL_DAY	No	Yes	No	No
REPL_ITEM_LOC_ UPDATES	No	Yes	No	No
SVC_LIKE_STORE_ STAGING	Yes	Yes	No	Yes
SVC_LIKE_STORE_GTT	Yes	Yes	No	Yes

Design Assumptions

N/A

straddbatch.ksh(Store Add Asynchronous Process)

Module Name	straddbatch.ksh
Description	Store Add Asynchronous Process
Functional Area	Foundation Data
Module Type	Admin
Module Technology	.ksh
Catalog ID	RMS496
Runtime Parameters	N/A

Business Overview

This asynchronous process creates new stores in Merchandising, along with all their associated records when a new store is initiated online in Merchandising or via the Store Subscription API.

Key Tables Affected

Table 3–24 Key Tables Affected

Table	Select	Insert	Update	Delete
STORE _ADD	Yes	No	No	Yes
STORE	Yes	Yes	No	No
STOCK_LEDGER_INSERTS	No	Yes	No	No
RPM_ZONE	No	Yes	No	No
RPM_ZONE_LOCATION	No	Yes	No	No
RMS_ASYNC_STATUS	Yes	Yes	Yes	No
RMS_ASYNC_RETRY	Yes	Yes	Yes	No
RMS_ASYNC_JON	Yes	No	No	No

Table 3–24 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
LOC_TRAITS_MATRIX	No	Yes	No	No
COST_ZONE	No	Yes	No	No
COST_ZONE_GROUP_LOC	No	Yes	No	No
STORE_HIERARCHY	No	Yes	No	No
WF_COST_RELATIONSHIP	No	Yes	No	No
SOURCE_DLVRY_SCHED	No	Yes	No	No
SOURCE_DLVRY_SCHED_EXC	No	Yes	No	No
SOURCE_DLVRY_SCHED_DAYS	No	Yes	No	No
COMPANY_CLOSED_EXCEP	No	Yes	No	No
LOCATION_CLOSED	No	Yes	No	No
POS_STORE	No	Yes	No	No
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
STORE_ADD_L10N_EXT	Yes	Yes	No	Yes
STORE_ADD_CFA_EXT	Yes	Yes	No	Yes

Design Assumptions

The materialized views MV_LOC_SOB, MV_L10N_ENTITY and MV_LOC_PRIM_ADDR will be refreshed after the store has been added. It is assumed that the materialized view will still be available to other processes during the refresh.

Queue Creation

The function RMS_ASYNC_QUEUE_SQL.CREATE_QUEUE_SUBSCRIBER is called to drop and recreate the queue table if one already exists. This function is called with the JOB_TYPE as STORE_ADD (for example, the constant ASYNC_JOB_STORE_ADD) to create a queue for store processing.

Design Overview - Process Steps

This section describes the key design aspect of the store add process.

The overall process consists of 3 steps as outlined below.

1. New (status-code: 00NEW). This is the status when store is just created.
2. Store-Add (status-code: 01STOREADD)
3. Store-Add-Post (status-code: 02STOREADD_POST)

The status-code of the current completed step of the process is updated in store_add.process_status column.

If STORE_ADD.LIKESTORE column is not null for the store, the status will remain in 02STOREADD_POST and the record will be picked up by the likestorebatch.ksh which

runs as an hourly job. If not, then the STORE entry will be removed from the STORE_ADD table.

Running entire store-add as batch in case of AQ issues

In case of Oracle AQ issues if store-add step is not running in async mode then entire store-add proess can also be run in batch using below command

```
storeaddbatch.ksh $UP
```

This is provided only as a workaround in case of AQ issues. The recommended method is to let store-add step be processed in Async through AQ as it is designed.

Building Schedule Dependencies between Async process and other batches

Customers may need to build scheduling dependencies between async processes and other batch programs. For example, making pos-extract batches dependent upon completion of Like-store step of the store-add process. To do that, create a job in scheduler using following command and make required batches dependent upon this job.

```
straddasyncwait.ksh $UP "03LIKESTORE"
```

Similarly, if batch program needs to be made dependent upon other steps, schedule jobs by passing desired status.

Monitoring Progress of Store-Add Processes

The current completed step of the store-add process is updated in store_add.process_status column. In case of a Like-Store step (which is a separate batch program) the status of a store will remain in 02STOREADD_POST, until it is processed by the likestore batch program, which will in turn change the status to 03LIKETORE.

Once the process is completed, the store will be subsequently removed from the STORE_ADD table. If not, then the status will be changed to '05LIKESTORE_FAIL'.

CORESVC_STORE_ADD_SQL. ADD_STORE (Store Add Asynchronous Process)

Module Name	CORESVC_STORE_ADD_SQL. ADD_STORE
Description	Asynchronous Process
Functional Area	Foundation Data
Module Type	Admin
Module Technology	PL SQL
Catalog ID	RMS496
Runtime Parameters	N/A

Business Overview

This asynchronous process creates new stores in Merchandising, along with all their associated records when a new store is initiated online in Merchandising or via the Store Subscription API. Previously, the likestore functionality is also processed within the store add asynchronous process, but this has now been decoupled from the store

add program and now runs as a separate hourly batch job, removing the dependency between both processes.

Key Tables Affected

Table 3–25 Key Tables Affected

Table	Select	Insert	Update	Delete
STORE _ADD	Yes	No	No	Yes
STORE	Yes	Yes	No	No
STOCK_LEDGER_INSERTS	No	Yes	No	No
RPM_ZONE	No	Yes	No	No
RPM_ZONE_LOCATION	No	Yes	No	No
RMS_ASYNC_STATUS	Yes	Yes	Yes	No
RMS_ASYNC_RETRY	Yes	Yes	Yes	No
RMS_ASYNC_JON	Yes	No	No	No
LOC_TRAITS_MATRIX	No	Yes	No	No
COST_ZONE	No	Yes	No	No
COST_ZONE_GROUP_ LOC	No	Yes	No	No
STORE_HIERARCHY	No	Yes	No	No
WF_COST_RELATIONSHIP	No	Yes	No	No
SOURCE_DLVRY_SCHED	No	Yes	No	No
SOURCE_DLVRY_SCHED_ EXC	No	Yes	No	No
SOURCE_DLVRY_SCHED_ DAYS	No	Yes	No	No
COMPANY_CLOSED_ EXCEP	No	Yes	No	No
LOCATION_CLOSED	No	Yes	No	No
POS_STORE	No	Yes	No	No
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
STORE_ADD_L10N_EXT	Yes	Yes	No	Yes
STORE_ADD_CFA_EXT	Yes	Yes	No	Yes

Design Assumptions

The materialized views MV_LOC_SOB, MV_L10N_ENTITY and MV_LOC_PRIM_ ADDR will be refreshed after the store has been added. It is assumed that the materialized view will still be available to other processes during the refresh.

Queue Creation

The function RMS_ASYNC_QUEUE_SQL.CREATE_QUEUE_SUBSCRIBER is called to drop and recreate the queue table if one already exists. This function is called with the

JOB_TYPE as STORE_ADD (for example, the constant ASYNC_JOB_STORE_ADD) to create a queue for store processing.

Design Overview - Process Steps

This section describes the key design aspect of the store add process.

The overall process consists of 3 steps as outlined below.

1. New (status-code: 00NEW). This is the status when store is just created.
2. Store-Add (status-code: 01STOREADD)
3. Store-Add-Post (status-code: 02STOREADD_POST)

The status-code of the current completed step of the process is updated in store_add.process_status column.

If STORE_ADD.LIKESTORE column is not null for the store, the status will remain in 02STOREADD_POST and the record will be picked up by the likestorebatch.ksh which runs as an hourly job. If not, then the STORE entry will be removed from the STORE_ADD table.

Package Impact

Package name: coresvc_store_add_sql

Spec file name: coresvc_store_adds.pls

File name: coresvc_store_adds/b.pls

Function Level Description - ADD_STORE

```
Function: ADD_STORE
           (O_error_message          IN OUT  RTK_ERRORS.RTK_TEXT%TYPE,
            I_rms_async_id           IN      RMS_ASYNC_STATUS.RMS_ASYNC_ID%TYPE)
```

This function contains the core logic for adding a new store to Merchandising. The process of adding a store to Merchandising starts with store.fmb form. When a user creates a new store by using the form, an entry is made in the STORE_ADD table. Also entries are made into RMS_ASYNC_STATUS with the status as new and RMS_ASYNC_RETRY tables with a new RMS_ASYNC_ID. The RMS_ASYNC_ID is placed in the queue for processing. The de-queue process picks the RMS_ASYNC_ID generated and based on the JOB_TYPE (STORE_ADD) calls the CORESVC_STORE_ADD_SQL.ADD_STORE for further processing.

This function:

- Calls PM_NOTIFY_API_SQL.NEW_LOCATION to create pricing records to update the Pricing tables.
- Calls the functions L10N_FLEX_ATTRIB_SQL.ADD_STORE_ATTRIB and CFA_SQL.ADD_STORE_ATTRIB.
- Makes entries into cost-zone tables.
- If like-store is mentioned and delivery schedule needs to be copied then copy source-delivery-schedule information. Hence entries are made into SOURCE_DLVRY_SCHED, SCHED_EXC and SCHED_DAYS tables.
- If like-store is mentioned and locations close information needs to be copied then make entries into COMPANY_CLOSED_EXCEP and LOCATION_CLOSED tables based on like store.

- Calls the function STKLEDGR_SQL.STOCK_LEDGER_INSERT to make entry into STOCK_LEDGER_INSERTS table.
- Copies WF_COST_RELATIONSHIP and DEAL_PASSTHRU data for the specified costing location.
- If like-store is mentioned then call the local function LIKE_STORE.
- The MV_LOC_SOB, MV_L10N_ENTITY and MV_LOC_PRIM_ADDR materialized views are refreshed as well.
- After completion of the process, it deletes the records from STORE_ADD_L10N_EXT, STORE_ADD_CFA_EXT and STORE_ADD tables.

On successful creation of the store you are prompted with a message saying the RMS_ASYNC_ID is processed successfully. In case there is a failure during the store creation you will also be notified. You have to use the Asynchronous Job log form to view and reprocess the failed store. On clicking on reprocess in the Asynchronous Job log form an entry is made into the RMS_ASYNC_RETRY table. The RMS_ASYNC_ID is again placed in the queue for processing.

Spec file name: rmsasyncprocs/b.pls

Function Level Description - ENQUEUE_STORE_ADD

```
Function: ENQUEUE_STORE_ADD
           (O_error_message          IN OUT  RTK_ERRORS.RTK_TEXT%TYPE,
            I_rms_async_id           IN      RMS_ASYNC_STATUS.RMS_ASYNC_ID%TYPE)
```

This function adds the RMS_ASYNC_ID associated with the JOB_TYPE STORE_ADD created from the store form to the asynchronous queue. It also makes entries into the RMS_ASYNC_STATUS and RMS_ASYNC_RETRY table to track the status of the asynchronous job.

Function Level Description - ENQUEUE_STORE_ADD_RETRY

```
Function: ENQUEUE_STORE_ADD_RETRY
           (O_error_message          IN OUT  RTK_ERRORS.RTK_TEXT%TYPE,
            I_rms_async_id           IN      RMS_ASYNC_STATUS.RMS_ASYNC_ID%TYPE)
```

This function puts the RMS_ASYNC_ID associated with a STORE_ADD event to the asynchronous queue again for re-processing. It is invoked through the asynchronous job log form.

Function Level Description - NOTIFY_STORE_ADD

```
Procedure: NOTIFY_STORE_ADD(context     raw,
                            reginfo     sys.aq$_reg_info,
                            descr       sys.aq$_descriptor,
                            payload     raw,
                            payloadl    number)
```

This procedure is called during the de-queue process. This procedure calls the function CORESVC_STORE_ADD_SQL.ADD_STORE for store creation. Once the store creation is completed successfully it calls the function RMS_ASYNC_PROCESS_SQL.WRITE_SUCCESS to update the status of the RMS_ASYNC_ID as success. During a failure in store creation it calls the function RMS_ASYNC_PROCESS_SQL.WRITE_ERROR to update the status as error and also to update the error message. You are notified of the success/failure of the store creation process.

Operations and Monitoring

This section describes the details required for running and monitoring this process.

Running entire Store-Add as Batch in Case of AQ Issues

In case of Oracle AQ issues if a store-add step is not running in async mode then the entire store-add process can also be run in batch using below command.

```
storeaddbatch.ksh $UP
```

This is provided only as a workaround in case of AQ issues. The recommended method is to let the store-add step be processed in Async through AQ as it is designed.

Building Schedule Dependencies between Async Process and other Batches

Customers may need to build scheduling dependencies between async processes and other batch programs. For example, making pos-extract batches dependent upon completion of a Like-store step of the store-add process. To do that, create a job in the scheduler by using the following command and make the required batches dependent upon this job.

```
straddasyncwait.ksh $UP "03LIKESTORE"
```

Similarly, if the batch program needs to be made dependent upon other steps, schedule jobs by passing desired status.

Monitoring Progress of Store-Add Processes

The current completed step of the store-add process is updated in the store_add.process_status column. In case of a Like-Store step (which is a separate batch program), the status of a store will remain in 02STOREADD_POST, until it is processed by the likestore batch program, which will in turn change the status to 03LIKETORE.

Once the process is completed, the store will be subsequently removed from the STORE_ADD table. If not, then the status will be changed to '05LIKESTORE_FAIL'.

4

Item Maintenance

This chapter contains information about the batch processes that relate to item maintenance. These processes include general item integration and processes to make mass changes to low level item information.

Program Summary

Table 4–1 Item Maintenance - Program Summary

Program	Description
sitmain.pc	Scheduled Item Maintenance
vatdlxpl.pc	Mass VAT Updates for Items/Locations
itm_indctn_purge.ksh	Purge Item induction staging tables
item_loc_purge_job	Daily Purge of Item-Location Data

sitmain (Scheduled Item Maintenance)

Module Name sitmain.pc

Description Scheduled Item Maintenance

Functional Area Item Maintenance

Module Type Business Processing

Module Technology ProC

Catalog ID RMS357

Wrapper Script rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Scheduled item maintenance is a method of performing mass changes on item/location information. Scheduled item maintenance uses item and location lists to make the process of changing lots of information very easy for end users.

This program explodes the intersection of these items and location lists to make the scheduled changes at the specific item/location level.

Restart/Recovery

This program has inherent restart ability because records are deleted from the scheduled item detail table as they are processed. The logical unit of work is an item/location combination.

Key Tables Affected

Table 4–2 Key Tables Affected

Table	Select	Insert	Update	Delete
SIT_EXPLODE	Yes	No	Yes	No
SIT_DETAIL	Yes	No	No	Yes
ITEM_LOC	Yes	Yes	Yes	No
MC_REJECTIONS	Yes	No	Yes	No
ITEM_MASTER	Yes	No	No	No
PRICE_HIST	No	Yes	No	No
ITEM_LOC_SOH	No	Yes	No	No

Design Assumptions

N/A

vatdlxpl (Mass VAT Updates for Items/Locations)

Module Name	vatdlxpl.pc
Description	Mass VAT Updates for Items/Locations
Functional Area	Item Maintenance
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS384
Runtime Parameters	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program updates VAT information for each item associated with a given VAT region and VAT code.

Restart/Recovery

This batch program performs commits to the database for every commit max number of rows.

Key Tables Affected

Table 4–3 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
VAT_CODE_RATES	Yes	No	No	No
VAT_ITEM	Yes	Yes	Yes	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPP_COUNTRY_LOC	Yes	No	Yes	No
ITEM_LOC	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
STORE	Yes	No	No	No
CLASS	Yes	No	No	No

Design Assumptions

N/A

itm_indctn_purge.ksh (Purge Item Induction Staging Tables)

Module Name	itm_indctn_purge.ksh
Description	Purge item induction staging tables
Functional Area	Foundation-Items
Module Type	Admin
Module Technology	Shell Script
Catalog ID	RMS498
Runtime Parameters	N/A

Design Overview

The purpose of this module is to remove old item records from the staging tables. Records that are candidates for deletion are:

- Processes that have successfully been processed or processed with warnings that have been uploaded to Merchandising or downloaded to S9T
- Processes that have status = 'PE', processed with errors and have no linked data
- Processes in error status where all other related records containing the process ID have been processed successfully

- Processes that have errors and are past the data retention days (system_options.proc_data_retention_days)
- All item records within a process where all related records for the item in the other staging tables are successfully uploaded to Merchandising. The process tracker record for that process should not be deleted if there are other item records that are not uploaded to Merchandising.

Scheduling Constraints

Table 4–4 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

Restart ability is implied, because the records that are selected from the cursor are deleted before the commit.

Key Tables Affected

Table 4–5

Table	Select	Insert	Update	Delete
PROC_DATA_RETENTION_DAYS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
SVC_PROCESS_TRACKER	Yes	No	No	Yes
SVC_PROCESS_ITEMS	No	No	No	Yes
SVC_ITEM_COST_DETAIL	No	No	No	Yes
SVC_ITEM_COST_HEAD	No	No	No	Yes
SVC_ITEM_COUNTRY	No	No	No	Yes
SVC_ITEM_COUNTRY_L10N_EXT	No	No	No	Yes
SVC_ITEM_MASTER	No	No	No	Yes
SVC_ITEM_MASTER_TL	No	No	No	Yes
SVC_ITEM_MASTER_CFA_EXT	No	No	No	Yes
SVC_ITEM_SUPPLIER	No	No	No	Yes
SVC_ITEM_SUPPLIER_TL	No	No	No	Yes
SVC_ITEM_SUPPLIER_CFA_EXT	No	No	No	Yes
SVC_ITEM_SUPP_COUNTRY	No	No	No	Yes
SVC_ITEM_SUPP_COUNTRY_CFA_EXT	No	No	No	Yes

Table 4–5 (Cont.)

Table	Select	Insert	Update	Delete
SVC_ITEM_SUPP_COUNTRY_DIM	No	No	No	Yes
SVC_ITEM_SUPP_MANU_COUNTRY	No	No	No	Yes
SVC_ITEM_SUPP_UOM	No	No	No	Yes
SVC_ITEM_XFORM_DETAIL	No	No	No	Yes
SVC_ITEM_XFORM_HEAD	No	No	No	Yes
SVC_ITEM_XFORM_HEAD_TL	No	No	No	Yes
SVC_PACKITEM	No	No	No	Yes
SVC_RPM_ITEM_ZONE_PRICE	No	No	No	Yes
SVC_XITEM_RIZP_LOCS	No	No	No	Yes
SVC_XITEM_RIZP	No	No	No	Yes
SVC_ITEM_SEASONS	No	No	No	Yes
SVC_UDA_ITEM_DATE	No	No	No	Yes
SVC_UDA_ITEM_FF	No	No	No	Yes
SVC_UDA_ITEM_LOV	No	No	No	Yes
SVC_VAT_ITEM	No	No	No	Yes
SVC_ITEM_IMAGE	No	No	No	Yes
SVC_ITEM_IMAGE_TL	No	No	No	Yes
SVC_ITEM_HTS	No	No	No	Yes
SVC_ITEM_HTS_ASSESS	No	No	No	Yes
SVC_COST_SUSP_SUP_HEAD	No	No	No	Yes
SVC_COST_SUSP_SUP_DETAIL_LOC	No	No	No	Yes
SVC_COST_SUSP_SUP_DETAIL	No	No	No	Yes
SVC_CFA_EXT	No	No	No	Yes
CORESVC_ITEM_ERR	No	No	No	Yes
S9T_ERRORS	No	No	No	Yes
SVC_PROCESS_CHUNKS	No	No	No	Yes
S9T_FOLDER	No	No	No	Yes

item_loc_purge_job (Daily Purge of Item-Location Data)

Module Name	item_loc_purge_job
Description	Daily Purge of Item-Location Data
Functional Area	Administration
Module Type	Admin - Adhoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

When users 'delete' an item-location record in the RMS user interface, information is generally not immediately deleted at the database level; instead, data is marked as being in deleted status and also inserted into the DAILY_PURGE table.

Thread assignment program (ITEM_LOC_PURGE_THREAD) will filter eligible records from daily purge (DAILY_PURGE) table wherein all entities ready for purging are exclusively related to Item-Location (ITEM_LOC table) records. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_ITEM_LOC_PURGE_STG.

The Business logic program (ITEM_LOC_PURGE) will process all records from the staging table. Using bulk processing, this program will delete item-location data from item-location related and associated tables. Complex referential integrity relationships determine whether data can actually be deleted from the database. This program checks these complex rules. If the deletion request passes the rules, this job will continues to delete the data. If it is not able to delete the data, it writes a record to the DAILY_PURGE_ERROR_LOG table for further investigation. This program will continue to attempt to delete marked data until all references have been purged from the system and the deletion of the item-location data finally succeeds. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 4–6 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 4–7

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No

Table 4–7 (Cont.)

Table	Select	Insert	Update	Delete
B8D_ITEM_LOC_PURGE_STG	Yes	No	No	No
DAILY_PURGE	Yes	No	No	Yes
DAILY_PURGE_ERROR_LOG	Yes	Yes	No	Yes
REPL_RESULTS	Yes	No	No	Yes
BUYER_WKSHT_MANUAL	Yes	No	No	Yes
SUB_ITEMS_DETAIL	Yes	No	No	Yes
SUB_ITEMS_HEAD	Yes	No	No	Yes
REPL_ATTR_UPDATE_EXCLUDE	Yes	No	No	Yes
MASTER_REPL_ATTR	Yes	No	No	Yes
REPL_DAY	Yes	No	No	Yes
REPL_ITEM_LOC	Yes	No	No	Yes
REPL_ITEM_LOC_UPDATES	Yes	Yes	No	Yes
REPL_ITEM_LOC_SUPP_DIST	Yes	No	No	Yes
COST_SUSP_SUP_DETAIL_LOC	Yes	No	No	Yes
FUTURE_COST	Yes	No	No	Yes
ITEM_LOC_MFQUEUE	Yes	No	No	Yes
ITEM_LOC	Yes	No	No	Yes
ITEM_LOC_SOH	Yes	No	No	Yes
ITEM_LOC_TRAITS	Yes	No	No	Yes
ITEM_LOC_CFA_EXT	Yes	No	No	Yes
ITEM_SUPP_COUNTRY_BRACKET_ COST	Yes	No	No	Yes
ITEM_SUPP_COUNTRY_LOC_CFA_EXT	Yes	No	No	Yes
ITEM_SUPP_COUNTRY_LOC	Yes	No	No	Yes

I/O Specification

NA

5

Purchase Order

Merchandising is the system of record in the Oracle Retail Suite for Purchase Orders (POs). Purchase orders can be created through the Merchandising UI, integration with products such as *Oracle Retail Advanced Inventory Planning* or integration with other 3rd party systems. Once purchase orders are created in Merchandising, there are a number of batch processes that manage PO data.

Batch Design Summary

The following batch designs are included in this functional area:

- edidlord.pc (Download of Purchase Order from Merchandising to Suppliers)
- ediupack.pc (Upload Purchase Order and Purchase Order Change Acknowledgements from Suppliers to Merchandising)
- vrplbld.pc (Build Purchase Orders for Vendor Generated Orders)
- genpreiss.pc (Generate Pre-Issued Order Numbers)supcnstr.pc (Scale Purchase Orders Based on Supplier Constraints)
- supcnstr.pc (Scale Purchase Orders Based on Supplier Constraints)
- orddscnt.pc (Apply Deal Discounts to Purchase Orders)
- ordupd.pc (Update Retail Values on Open Purchase Orders)
- ordautcl.pc (Auto Close Purchase Orders)
- order_auto_close_job (Auto Close Purchase Orders)
- ordrev.pc (Write Purchase Order Information to Purchase Order History Tables)
- order_revision_job (Write Purchase Order Information to Purchase Order History Tables)
- ordprg.pc (Purge Aged Purchase Orders)
- order_purge_job (Purge Aged Purchase Orders)
- poindbatch.ksh(Upload of PO induction data through batch)
- po_indctn_purge.ksh(Purge data from PO induction staging tables)

edidlord (Download of Purchase Orders from Merchandising to Suppliers)

Module Name edidlord.pc

Description	Download of Purchase Order from Merchandising to Suppliers
Functional Area	Purchase Order
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS46
Wrapper Script	rmswrap_multi_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Orders created within the Oracle Retail system are written to a flat file if they are approved and marked as EDI orders. This module is used to write new and changed purchase order data to a flat file in the Oracle Retail standard format. The translation to EDI format is expected to take place via a 3rd party translation utility. The order revision tables and allocation revision tables are also used to ensure that the latest changes are being sent and to allow both original and modified values to be sent. These revision tables are populated during the online ordering process and the batch replenishment process whenever an order has been approved, and constitutes a history of all revisions to the order.

The program sums up all quantities to the physical warehouse level from the virtual warehouse level for an order, before writing it into the output file.

If shipments are to be pre-marked by the supplier for cross docking, then along with the order information: allocation, location and quantities are also sent.

If the backhaul type is specified as “Calculated”, then the backhaul allowances will be calculated.

If the order contains pack items; hierarchical pack information is sent (this may include outer packs, inner packs, and fashion styles with associated pack templates as well as component item information).

If the order is a Drop Ship Customer Order (location is a non-stockholding store), the customer billing and delivery information will be written to the flat file.

Restart/Recovery

The logical unit of work for this program is set at the supplier level. Threading is performed by the supplier using the v_restart_supplier view.

Restart ability is implied because the program updates ordhead.edi_sent_ind as records and are written out. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of the file I/O. The recommended commit counter setting is 10000 records.

I/O Specification

Integration Type Download from Merchandising

File Name	Determined by runtime parameter
Integration Contract	IntCon000012

Output File Layout

Table 5–1 File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	File head marker
	Line id	Number(10)	0000000001	Unique line id
	Translator id	Char(5)	DLORD	Identifies transaction type
	File create date	Char(14)	N/A	Vdate in YYYYMMDDHH24MISS format
TORDR	Record descriptor	Char(5)	TORDR	Order header information
	Line id	Number(10)	N/A	Unique file line id
	Transaction id	Number(10)	N/A	Unique transaction id
	Order change type	Char(2)	N/A	'CH' (changed) or 'NW' (new)
	Order number	Number(12)	N/A	Internal Oracle Retail order no
	Supplier	Number(10)	N/A	Internal Oracle Retail supplier id
	Vendor order id	Char(15)	N/A	External vendor_order_no (if available)
	Order written date	Char(14)	N/A	Order created date in YYYYMMDDHH24MISS format
	Original order approval date	Char(14)	N/A	Original order approval date in YYYYMMDDHH24MISS format
	Old Currency Code	Char(3)	N/A	Old order currency_code (ISO standard)
	New Currency Code	Char(3)	N/A	Changed order currency_ code (ISO standard)
	Old Shipment Method of Payment	Char(2)	N/A	Old ship_pay_method
	New Shipment Method of Payment	Char(2)	N/A	Changed ship_pay_method

Table 5–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Old Transportation Responsibility	Char(2)	N/A	Old fob_trans_res
	Old Transportation Responsibility Description	Char(250)	N/A	Old fob_trans_res_desc
	New Transportation Responsibility	Char(2)	N/A	Changed fob_trans_res
	New Trans. Resp. Description	Char(250)	N/A	New fob_trans_res_desc
	Old Title Passage Location	Char(2)	N/A	Old fob_title_pass
	New Title Passage Location	Char(2)	N/A	Changed fob_title_pass
	Old Title Passage Description	Char(250)	N/A	Old fob_title_pass_desc
	New Title Passage Description	Char(250)	N/A	Changed fob_title_pass_ desc
	Old not before date	Char(14)	N/A	Old not_before_date in YYYYMMDDHH24MISS format
	New not before date	Char(14)	N/A	Changed not_before_date in YYYYMMDDHH24MISS format
	Old not after date	Char(14)	N/A	Old not_after_date in YYYYMMDDHH24MISS format
	New not after date	Char(14)	N/A	Changed not_after_date in YYYYMMDDHH24MISS format
	Old Purchase type	Char(6)	N/A	Old Purchase type
	New Purchase type	Char(6)	N/A	New Purchase type
	Backhaul allowance	Char(20)	N/A	Backhaul allowance
	Old terms description	Char(240)	N/A	Old terms description from terms table

Table 5–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	New terms description	Char(240)	N/A	New terms description from terms table
	Old pickup date	Char(14)	N/A	Old pickup date YYYYMMDDHH24MISS
	New pickup date	Char(14)	N/A	New pickup date YYYYMMDDHH24MISS
	Old ship method	Char(6)	N/A	Old ship method
	New ship method	Char(6)	N/A	New ship method
	Old comment description	Char(2000)	N/A	Old comment description
	New comment description	Char(2000)	N/A	New comment description
	Supplier DUNS number	Char(9)	N/A	Supplier DUNS number
	Supplier DUNS location	Char(4)	N/A	Supplier DUNS location
	Customer order number	Char(48)	N/A	Master customer order number from the Order Management System
TITEM	File record descriptor	Char(5)	TITEM	Item info
	Line id	Number(10)	N/A	Unique line id
	Transaction id	Number(10)	N/A	Unique transaction id
	Item Number Type	Char(6)	N/A	Item_number_type
	Item	Char(25)	N/A	Item (For a pack item, this will be the pack number)
	Old Ref Item Number type	Char(6)	N/A	Item_number_type for old ref_item
	Old Ref Item	Char(25)	N/A	Old Ref_Item
	New Ref Item Number type	Char(6)	N/A	Item_number_type for new ref_item
	New Ref Item	Char(25)	N/A	Changed Ref_Item
	Vendor catalog number	Char(30)	N/A	Supplier_item (VPN)
	Free Form Description	Char(250)	N/A	Item_desc

Table 5–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Supplier Diff 1	Char(120)	N/A	Supplier's diff 1
	Supplier Diff 2	Char(120)	N/A	Supplier's diff 2
	Supplier Diff 3	Char(120)	N/A	Supplier's diff 3
	Supplier Diff 4	Char(120)	N/A	Supplier's diff 4
	Pack Size	Number(12)	N/A	Supplier defined pack size * 10000 (4 implied decimal places)
	Item line no	Number(10)	N/A	Indicates the detail item line number.
TPACK	File record descriptor	Char(5)	TPACK	Pack component info
	Line id	Number(10)	N/A	Unique line id
	Transaction id	Number(10)	N/A	Unique transaction id
	Pack id	Char(25)	N/A	Packitem_breakout.pack_no (same as item for the pack item)
	Inner pack id	Char(25)	N/A	Inner pack identification
	Pack Quantity	Number(12)	N/A	Packitem_breakout.pack_item_qty*10000 (4 implied decimal places)
	Component Pack Quantity	Number(12)	N/A	Packitem_breakout.comp_pack_qty*10000 (4 implied decimal places)
	Item Parent Part Quantity	Number(12)	N/A	Packitem_breakout.item_parent_pt_qty*10000 (4 implied decimal places)
	Item Quantity	Number(12)	N/A	Packitem_breakout.item_qty*10000 (4 implied decimal places)
	Item Number Type	Char(6)	N/A	Item number type
	Item	Char(25)	N/A	Item
	Ref Item Number Type	Char(6)	N/A	Ref_item_number_type
	Ref Item	Char(25)	N/A	Ref_item
	VPN	Char(30)	N/A	Supplier item (vpn)
	Supplier Diff 1	Char(120)	N/A	Supplier's diff 1
	Supplier Diff 2	Char(120)	N/A	Supplier's diff 2

Table 5–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Supplier Diff 3	Char(120)	N/A	Supplier's diff 3
	Supplier Diff 4	Char(120)	N/A	Supplier's diff 4
	Item Parent	Char(25)	N/A	Required when Pack Template is not NULL
	Pack template	Number(8)	N/A	Pack template associated w/style (packitem_ breakout.pack_tmpl_id)
	Template description	Char(250)	N/A	Description of pack template. sups_pack_tmpl_ desc.supp_pack_desc
TSHIP	Record type	Char(5)	TSHIP	Describes the file record-shipment information
	Line id	Number(10)	N/A	Unique file line number
	Transaction id	Number(10)	N/A	Unique transaction number
	Location type	Char(2)	N/A	'ST' store or 'WH' warehouse
	Ship to location	Number(10)	N/A	Location value form ordloc (store or warehouse – For warehouse,if multichannel option is ON, physical warehouse value is taken from warehouse)
	Old unit cost	Number(20)	N/A	Old unit cost*10000 (4 implied decimal places)
	New unit cost	Number(20)	N/A	New unit cost*10000 (4 implied decimal places)
	Old quantity	Number(12)	N/A	Old qty_ordered *10000 or qty_allocated*10000 (4 implied decimal places)
	New quantity	Number(12)	N/A	Changed qty_ordered*10000 or qty_allocated*10000 (4 implied decimal places)
	Old outstanding quantity	Number(12)	N/A	Old (qty_ordered-qty_ received)*10000 or (qty_ allocated-qty transferred)*10000 for an allocation (4 implied decimal places)
	New outstanding quantity	Number(12)	N/A	Changed qty_ordered-qty_ received (4 implied decimal places)(or qty_ allocated-qty_transferred, for an allocation)
	Cancel code	Char(1)	N/A	N/A

Table 5–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Old cancelled quantity	Number(12)	N/A	Previous quantity cancelled (4 implied decimal places)
	New cancelled quantity	Number(12)	N/A	Changed quantity cancelled (4 implied decimal places)
	Quantity type flag	Char(1)	N/A	'S'hip to 'A'llocate
	Store or warehouse indicator	Char(2)	N/A	'ST' (store) or 'WH' (warehouse)
	Old x-dock location	Number(10)	N/A	Alloc_detail location (store or wh)
	New x-dock location	Number(10)	N/A	Alloc_detail location (store or wh)
	Case length	Number(12)	N/A	Case length (4 implied decimal places)
	Case width	Number(12)	N/A	Case width (4 implied decimal places)
	Case height	Number(12)	N/A	Case height (4 implied decimal places)
	Case LWH unit of measure	Char(4)	N/A	Case LWH unit of measure
	Case weight	Number(12)	N/A	Case weight (4 implied decimal places)
	Case weight unit of measure	Char(4)	N/A	Case weight unit of measure
	Case liquid volume	Number(12)	N/A	Case liquid volume (4 implied decimal places)
	Case liquid volume unit of measure	Char(4)	N/A	Case liquid volume unit of measure
	Location DUNS number	Char(9)	N/A	Location DUNS number
	Location DUNS loc	Char(4)	N/A	Location DUNS loc
	Old unit cost init	Number(20)	N/A	Old unit cost init (4 implied decimal places)
	New unit cost init	Number(20)	N/A	New unit cost init (4 implied decimal places)
	Item/loc discounts	Number(20)	N/A	Item/loc discounts (4 implied decimal places)
TCUST	Record type	Char(5)	TCUST	Describes the file record-customer order information

Table 5–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Line id	Number(10)	N/A	Unique file line number
	Transaction id	Number(10)	N/A	Unique transaction number
	Delivery first name	Char(120)	N/A	First name for the delivery address on the order
	Delivery phonetic first name	Char(120)	N/A	Phonetic first name for the delivery address on the order
	Delivery last name	Char(120)	N/A	Last name for the delivery address on the order
	Delivery phonetic last name	Char(120)	N/A	Phonetic last name for the delivery address on the order
	Delivery preferred name	Char(120)	N/A	Preferred name for the delivery address on the order
	Delivery company name	Char(120)	N/A	Company name for the delivery address on the order
	Delivery address Line 1	Char(240)	N/A	First line of the delivery address of the customer
	Delivery address Line 2	Char(240)	N/A	Second line of the delivery address of the customer
	Delivery address Line 3	Char(240)	N/A	Third line of the delivery address of the customer
	Delivery county	Char(250)	N/A	County portion of the delivery address
	Delivery city	Char(120)	N/A	City portion of the delivery address
	Delivery state	Char(3)	N/A	State portion of the delivery address
	Delivery country ID	Char(3)	N/A	Country portion of the delivery address
	Delivery post	Char(30)	N/A	Postal code portion of the delivery address
	Delivery jurisdiction	Char(10)	N/A	Jurisdiction code of the delivery country-state relationship
	Delivery phone	Char(20)	N/A	Phone number in the delivery information
	Billing first name	Char(120)	N/A	First name for the billing address on the order
	Billing phonetic first name	Char(120)	N/A	Phonetic first name for the billing address on the order

Table 5–1 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Billing last name	Char(120)	N/A	Last name for the billing address on the order
	Billing phonetic last name	Char(120)	N/A	Phonetic last name for the billing address on the order
	Billing preferred name	Char(120)	N/A	Preferred name for the billing address on the order
	Billing company name	Char(120)	N/A	Company name for the billing address on the order
	Billing address Line 1	Char(240)	N/A	First line of the billing address of the customer
	Billing address Line 2	Char(240)	N/A	Second line of the billing address of the customer
	Billing address Line 3	Char(240)	N/A	Third line of the billing address of the customer
	Billing county	Char(250)	N/A	County portion of the billing address
	Billing city	Char(120)	N/A	City portion of the billing address
	Billing state	Char(3)	N/A	State portion of the billing address
	Billing country ID	Char(3)	N/A	Country portion of the billing address
	Billing post	Char(30)	N/A	Postal code portion of the billing address
	Billing jurisdiction	Char(10)	N/A	Jurisdiction code of the billing country-state relationship
	Billing phone	Char(20)	N/A	Phone number in the billing information
TTAIL	Record type	Char(5)	TTAIL	Describes file record – marks end of order
	Line id	Number(10)	N/A	Unique file line id
	Transaction id	Number(10)	N/A	Unique transaction id
	#Lines in transaction	Number(10)	N/A	Number of lines in transaction
FTAIL	Record type	Char(5)	FTAIL	Describes file record – marks end of file
	Line id	Number(10)	N/A	Unique file line id
	#lines	Number(10)	N/A	Total number of transaction lines in file (not including FHEAD and FTAIL)

For a new order, the "old" fields should be blank. For a changed order, both old and new fields should hold values. If the value has changed, "old" values come from the revision tables for the latest revision before the current one (the last one sent), while new orders come from the ordering tables.

- FHEAD - REQUIRED: File identification, one line per file.
- TORDR - REQUIRED: Order level information, one line per order.
- TITEM - REQUIRED: Item description, multiple lines per order possible.
- TPACK - OPTIONAL: Pack contents, multiple lines per order possible. This line will be written only for pack items.
- TSHIP - REQUIRED: Ship to location and quantity, allocation location, multiple lines per item possible. Allocation information is optional on this line-will exist if premark_ind is 'Y'.
- TCUST - OPTIONAL: Customer order information, one line per order. This line will be written only for Drop Ship Customer Orders.
- TTAIL - REQUIRED: Order end, one line per order.
- FTAIL - REQUIRED: End of file marker, one line per file.Output File Layout

Design Assumptions

N/A

ediupack (Upload Purchase Order and Purchase Order Change Acknowledgements from Suppliers to Merchandising)

Module Name	ediupack.pc
Description	Upload Purchase Order and Purchase Order Change Acknowledgements from Suppliers to Merchandising
Functional Area	Purchase Orders
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS48
Wrapper Script	rmswrap_in_rej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program has four functions:

1. to acknowledge vendor receipt of a buyer-generated order without changes,
2. to acknowledge vendor receipt of a buyer-generated order with date, cost or quantity modifications,
3. to notify buyer of a vendor-generated order, and

4. to acknowledge order cancellations.

All acknowledgements update the ORDHEAD table with acknowledgement information.

When the supplier sends the acknowledgement with modifications, they can send the entire purchase order or only the changes. The file details are matched to the current order. If the Not Before Date, Not After Date, Quantity, Price, and item all match the current order, then no changes were submitted. If one of the variables is blank, for example the price, assume that no pricing changes were made. As soon as one of the variables does not match, the order has been changed. These changes will not be written directly to the order; they will be written to the revision tables. Revisions will be accepted in the on-line ordering screens and changed orders will be resubmitted via EDIDLORD.

Vendor generated orders will create new orders by inserting new records on the EDI temporary order tables.

For Customer Order POs created through an external Order Management System (OMS) and Franchise Order POs, the modifications to the dates, quantity and cost are applied automatically (and will not need to be accepted online). Also, changes to Franchise POs through this program will not affect their associated Franchise orders.

Restart/Recovery

The files will not have enough volume to warrant the implementation of restart recovery for commit/rollback considerations but minimal file-based restart/recovery capability will be added. The logical unit of work is a complete transaction represented by detail lines between the transaction header and transaction tail.

A savepoint will be issued before each transaction header record is successfully processed. If a non-fatal error occurs, a rollback to the last savepoint will be issued so that the rejected records are not posted to the database. If a fatal error occurs and restart is necessary, processing will restart at the last commit point.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000014

Input File Layout

Table 5–2 ediupack - Input File

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File head descriptor	Char(5)	FHEAD	Describes file line type
	Line id	Number(10)	0000000001	Sequential file line number
	File Type Definition	Char(4)	ORAK	Identifies file as 'Order Acknowledgment Import'

Table 5–2 (Cont.) ediupack - Input File

Record Name	Field Name	Field Type	Default Value	Description
THEAD	File record descriptor	Char(5)	THEAD	Describes file line type
	Line id	Number(10)	Line number in file	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
	Acknowledge type	Char(2)	N/A	AP-product replenishment AK- Acknowledge or change CA-cancel order (no detail)
	Order number	Char(15)	N/A	May be external order number (vendor order number) OR Oracle Retail order number
	Written_date	Char(8)	N/A	Written date in YYYYMMDD format
	Supplier number	Number(10)	N/A	Supplier number
	Not before date	Char(8)	N/A	Not_before_date YYYYMMDD
	Not after date	Char(8)	N/A	Not_after_date YYYYMMDD
	Purchase type	Char(6)	N/A	Specifies type of purchase – may be blank
	Pickup date	Char(8)	N/A	Pickup_date YYYYMMDD – may be blank
TITEM	File record descriptor	Char(5)	TITEM	Describes file line type
	Line id	Number(10)	Line number in file	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
	ITEM	Char(25)	N/A	Item (either item or ref_item must be defined)
	Ref_item	Char(25)	N/A	Reference item (either item or ref_item must be defined)
	Vendor catalog number	Char(30)	N/A	VPN (Vendor Product Number)
	Unit cost value	Number(20)	N/A	Unit_cost * 10000 (4 implied decimal places)
	Loc_type	Char(2)	N/A	'ST' for store, 'WH' for warehouse
	Location	Number(10)	N/A	If NULL, apply to all locations for this item
	Pickup location	Char(250)	N/A	Location to pick up item – may be blank

Table 5–2 (Cont.) ediupack - Input File

Record Name	Field Name	Field Type	Default Value	Description
TSHIP	File record descriptor	Char(5)	TSHIP	Describes file line type
	Line id	Number(10)	Line number in file	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
	Store/wh indicator	Char(2)	N/A	'ST' for store, 'WH' for warehouse
	Ship to location	Number(10)	N/A	Store or warehouse number
	Quantity	Number(12)	N/A	Quantity ordered * 10000 (4 implied decimal places)
TTAIL	File record descriptor	Char(5)	TTAIL	Describes file line type
	Line id	Number(10)	Line number in file	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
	Lines in transaction	Number(6)	N/A	Total number of lines in this transaction
FTAIL	File record descriptor	Char(5)	FTAIL	Marks end of file
	Line id	Number(10)	Line number in file	Sequential file line number
	Number of transactions	Number(10)	´NA	Number of lines between FHEAD and FTAIL

Design Assumptions

N/A

vrplbld (Build Purchase Orders for Vendor Generated Orders)

Module Name	vrplbld.pc
Description	Build Purchase Orders for Vendor Generated Orders
Functional Area	Purchase Orders
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS387
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This purpose of this module is to continue the process started by the batch program ediupack.pc of building purchase orders that reflect the vendor-generated orders as received through the EDI 855. This module will process records from the temporary EDI order table and create the purchase orders on the PO tables.

The post-processing function of this batch on the prepost batch truncates the EDI temporary order table.

Restart/Recovery

The logical unit of work for the program is a vendor order number, department and supplier combination. The program's restartability is dependent on the value of the Department Level Orders system parameter. Allowing multi-department orders (that is, the indicator is set to 'N') will restart the program from the last successfully processed vendor order number and supplier. If the system requires a department on the orders (that is, the indicator is set to 'Y'), then the program will restart from the last successfully processed vendor order number, department, and supplier.

Key Tables Affected

Table 5–3 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_MASTER	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPP_COUNTRY_LOC	Yes	No	No	No
SUP_IMPORT_ATTR	Yes	No	No	No
SUPS	Yes	No	No	No
EDI_ORD_TEMP	Yes	No	No	No
WH	Yes	No	No	No
ORDSKU	Yes	Yes	Yes	No
ORDHEAD	Yes	Yes	Yes	No
ORDLOC	No	Yes	No	No
DEAL_CALC_QUEUE	Yes	Yes	Yes	No
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
PROCUREMENT_UNIT_OPTIONS	Yes	No	No	No
L10N_DOC_DETAILS_GTT	Yes	Yes	No	No
MV_L10N_ENTITY	Yes	No	No	No
COUNTRY_ATTRIB	Yes	No	No	No
L10N_PKG_CONFIG	Yes	No	No	No
TSFHEAD	Yes	No	No	No
ORDHEAD_L10N_EXT	No	Yes	No	No

Table 5–3 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
TSFHEAD_L10N_EXT	No	Yes	No	No
MRT_L10N_EXT	No	Yes	No	No
FM_SYSTEM_OPTIONS	Yes	No	No	No
REV_ORDERS	No	No	No	Yes
ORDLOC_REV	No	Yes	No	No
ORDSKU_REV	No	Yes	No	No
ORDHEAD_REV	Yes	Yes	No	No

Design Assumptions

N/A

genpreiss (Generate Pre-Issued Order Numbers)

Module Name	genpreiss.pc
Description	Generate Pre-Issued Order Numbers
Functional Area	Purchase Orders
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS237
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Based on records on the SUPP_PREISSUE table, this batch program reserves order numbers for suppliers that do Vendor Managed Inventory (VMI) by placing these pre-generated order numbers on the ORD_PREISSUE table.

Restart/Recovery

The logical unit of work for this program is set at thesupplier level, based on a single record from the SUPP_PREISSUE table. It uses v_restart_supplier to achieve restart/recovery.

The changes will be posted when the commit_max_ctr value is reached and the value of the counter is subject to change based on implementation. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O.

Design Assumptions

N/A

supcnstr (Scale Purchase Orders Based on Supplier Constraints)

Module Name	supcnstr.pc
Description	Scale Purchase Orders Based on Supplier Constraints
Functional Area	Purchase Orders
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS368
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program will process all orders eligible for scaling during the nightly replenishment run. The purpose of this program is to select all of the orders created by the replenishment programs which are eligible for scaling. Once selected, the program will serve as a wrapper program and send each order number into the supplier constraint scaling library to actually perform the scaling on the order.

The orders which will be eligible for scaling are as follows:

If due order processing was used, only orders with a written date of today, origin type of zero (0) (replenishment order), due order processing indicator of Yes, due order indicator of Yes and a scale order to constraint indicator of Y will be processed. This encompasses all due orders created by replenishment which have constraints associated with them.

If due order processing was not used, only orders with a written date of today, origin type of zero (0) (replenishment order), order approve indicator of Yes, status of 'W'orksheet, due order processing indicator of No, due order indicator of Yes, and a scale order to constraint indicator of Yes will be processed. This encompasses all approved orders created by replenishment which have constraints associated with them.

For Franchise POs, their associated Franchise Orders will be updated when quantities of the franchise POs are changed due to supplier constraint.

Restart/Recovery

The logic unit of work for this program is an order number.

Locking Strategy

This batch locks order inventory management and order header tables during day runs.

Key Tables Affected

Table 5–4 Key Tables Affected

Table	Select	Insert	Update	Delete
ORDHEAD	Yes	No	Yes	No
ORD_INV_MGMT	Yes	No	Yes	No
PERIOD	Yes	No	No	No
WF_ORDER_HEAD	Yes	No	No	No
WF_ORDER_DETAIL	Yes	No	Yes	No

Design Assumptions

N/A

orddscnt (Apply Deal Discounts to Purchase Orders)

Module Name	orddscnt.pc
Description	Apply Deal Discounts to Purchase Orders
Functional Area	Purchase Orders
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS283
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module applies deals to a purchase order by calculating the discounts and rebates that are applicable to a purchase order. It will fetch orders that need to be recalculated for cost from the DEAL_CALC_QUEUE table. Using the dealordlib shared library, it will update the unit cost and populate the ORDLOC_DISCOUNT and ORDHEAD_ DISCOUNT tables.

Restart/Recovery

This program has inherent restart ability, since records are deleted from deal_calc_ queue as they are processed. Recommended maximum commit counter is low.

Design Assumptions

N/A

ordupd (Update Retail Values on Open Purchase Orders)

Module Name	ordupc.pc
Description	Update Retail Values on Open Purchase Orders
Functional Area	Purchase Orders
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS287
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will be used to automatically change all retail prices on purchase orders when a retail price change is implemented for an item on the order with the status of 'Worksheet',' Submit' and 'Approve'.

Open to buy is updated to give a more accurate picture of the retail value of open orders if the order is 'Approved' and if the department calculate the OTB as retail.

Restart/Recovery

This program does not contain restart/recovery logic.

Design Assumptions

N/A

ordautcl (Auto Close Purchase Orders)

Module Name	ordautcl.pc
Description	Auto Close Purchase Orders
Functional Area	Purchase Orders
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS282
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to process POs that need to be deleted or closed that meet certain conditions. The criteria are as mentioned below:

Category 1

- The order is not in 'C'ompleted status and was previously approved.
- The number of days between the latest ship date and the current date is greater than the 'Approved PO Close Delay' system parameter.
- There are no open shipments for the order.
- End of week date should not be null.

Category 2

- The order is not in 'C'ompleted status and was previously approved.
- A specified amount of time ('Approved PO Close Delay' system parameter) after the not after date of the PO has passed.
- A specified amount of time ('Partially Received PO Close Delay' system parameter) after the not after date has passed.
- A specified amount of time ('Partially Received PO Close Delay' system parameter) after the expected receipt date (or shipped date if the expected date has not been captured) has passed.
- There are no open appointments in the system for the order.

Category 3

- The order has a status of worksheet or submitted, and the order has never been previously approved.
- The number of days between the current date and the order creation date is greater than the 'Worksheet PO Clean Up Delay' system parameter.
- The order is a manual order (not created by replenishment).
- End of week date should not be null.

Retrieved orders are subsequently processed based on their category:

1. Category 1 orders will be closed. Closing an order involves adjusting the order quantities, shipment quantities and OTB. Any allocation associated with the order will also be closed if it is released 'X' number of days before vdate. The 'X' number of days is defaulted from an external system and set on the Merchandising codes table for code_type 'DEFT'.
2. For Category 2 orders, orders will be closed if there are no pending receipts or if the 'Auto Close Partially Received' system indicator is set to 'Y'.
3. Category 3 orders will be deleted from the system.

Restart/Recovery

Restart recovery is implicit since the program purges and cancels records in the database one order at a time.

Design Assumptions

N/A

order_auto_close_job (Auto Close Purchase Orders)

Module Name	order_auto_close_job
Description	Auto Close Purchase Orders
Functional Area	Purchase Orders
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (ORDER_AUTO_CLOSE_THREAD) will filter eligible records from order header (ORDHEAD) table based on defined criteria. This program is used to process POs that need to be deleted or closed that meet certain conditions. The criteria are as mentioned below:

Category 1

- The order is not in 'C'ompleted status and was previously approved.
- The number of days between the latest ship date and the current date is greater than the 'Approved PO Close Delay' system parameter.
- There are no open shipments for the order.
- End of week date should not be null.

Category 2

- The order is not in 'C'ompleted status and was previously approved.
- A specified amount of time ('Approved PO Close Delay' system parameter) after the not after date of the PO has passed.
- A specified amount of time ('Partially Received PO Close Delay' system parameter) after the not after date has passed.
- A specified amount of time ('Partially Received PO Close Delay' system parameter) after the expected receipt date (or shipped date if the expected date has not been captured) has passed.
- There are no open appointments in the system for the order.

Category 3

- The order has a status of worksheet or submitted, and the order has never been previously approved.
- The number of days between the current date and the order creation date is greater than the 'Worksheet PO Clean Up Delay' system parameter.

- The order is a manual order (not created by replenishment).
- End of week date should not be null.

These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_ORDER_AUTO_CLOSE_STG.

The Business logic program (ORDER_AUTO_CLOSE) will process all records from the staging table. Using bulk processing, this program will process retrieved records based on their category:

1. Category 1 orders will be closed. Closing an order involves adjusting the order quantities, shipment quantities and OTB. Any allocation associated with the order will also be closed if it is released 'X' number of days before vdate. The 'X' number of days is defaulted from an external system and set on the RMS codes table for code_type 'DEFT'.
2. For Category 2 orders, orders will be closed if there are no pending receipts or if the 'Auto Close Partially Received' system indicator is set to 'Y'.
3. Category 3 orders will be deleted from the system.

It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 5–5 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 1 hour interval - 12 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Order number

Restart/Recovery

NA

Key Tables Affected

Table 5–6 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_ORDER_AUTO_CLOSE_STG	Yes	No	Yes	Yes
ORDHEAD	Yes	No	Yes	Yes

Table 5–6 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SHIPMENT	Yes	No	Yes	No
APPT_HEAD	Yes	No	No	No
APPT_DETAIL	Yes	No	No	No
SHIPSKU	Yes	No	Yes	No
ORDLOC	No	No	Yes	Yes
ALLOC_DETAIL	No	No	Yes	Yes
OBLIGATION_COMP	No	No	No	Yes
WO_DETAIL	No	No	No	Yes
WO_HEAD	No	No	No	Yes
WO_SKU_LOC	No	No	No	Yes
WO_WIP	No	No	No	Yes
ALLOC_CHRG	No	No	No	Yes
ALLOC_HEADER	No	No	No	Yes
ORDLOC_DISCOUNT	No	No	No	Yes
TIMELINE	No	No	No	Yes
ORDSKU_TEMP	No	No	No	Yes
ORDLOC_TEM	No	No	No	Yes
ALLOC_CHRG_TEMP	No	No	No	Yes
ALLOC_DETAIL_TEMP	No	No	No	Yes
ALLOC_HEADER_TEMP	No	No	No	Yes
ORDLOC_EXP_TEMP	No	No	No	Yes
ORDSKU_HTS_ASSESS_ TEMP	No	No	No	Yes
ORDSKU_HTS_TEMP	No	No	No	Yes
ORDLOC_DISCOUNT_ TEMP	No	No	No	Yes
TIMELINE_TEMP	No	No	No	Yes
REQ_DOC_TEMP	No	No	No	Yes
WO_DETAIL_TEMP	No	No	No	Yes
WO_HEAD_TEMP	No	No	No	Yes
ORDLOC_WKSHT	No	No	No	Yes
ORDLOC_REV	No	No	No	Yes
ORDSKU_REV	No	No	No	Yes
ORDSKU	No	No	No	Yes
ORDCUST	No	No	No	Yes
ORDHEAD_REV	No	No	No	Yes
ORDLC	No	No	No	Yes
DEAL_COMP_PROM	No	No	No	Yes

Table 5–6 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
DEAL_ITEMLOC	No	No	No	Yes
DEAL_THRESHOLD	No	No	No	Yes
DEAL_DETAIL	No	No	No	Yes
DEAL_QUEUE	No	No	No	Yes
DEAL_CALC_QUEUE	No	No	No	Yes
DEAL_HEAD	No	No	No	Yes
ORD_INV_MGMT	No	No	No	Yes
REPL_RESULTS	No	No	No	Yes
REV_ORDERS	No	No	No	Yes
REQ_DOC	No	No	No	Yes
ORD_PREISSUE	No	No	No	Yes

Design Assumptions

NA

ordrev (Write Purchase Order Information to Purchase Order History Tables)

Module Name	ordrev.pc
Description	Write Purchase Order Information to Purchase Order History Tables
Functional Area	Purchase Orders
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS286
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Ordrev.pc will write versions of approved orders to the order revision history tables. When orders are approved or when approved orders are modified, this program selects order numbers from the REV_ORDERS table and writes current order information to the order/allocation revision tables. After the new version has been written to the order revision tables, all records will be deleted from the REV_ORDERS table for that order_no.

This program processes order changes made by the client that may need to be sent to the vendor. The order changes should always be referred to as ‘versions’ and kept

clearly distinct from order ‘revisions’ which are vendor changes uploaded via the ediupack program.

If an order is not in approved status at the time the batch program runs, then none of the above processing will occur. These records will stay on the REV_ORDERS table until the PO is approved or deleted.

Restart/Recovery

Restart ability will be implied because the records that are selected from the driving cursor will be deleted before the commit. Restart library functions will still be included to ensure that rollback segments are not exceeded (by committing at intervals) and to perform basic record keeping functionality.

Design Assumptions

N/A

order_revision_job (Write Purchase Order Information to Purchase Order History Tables)

Module Name	order_revision_job
Description	Write Purchase Order Information to Purchase Order History Tables
Functional Area	Purchase Orders
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Order changes made by the client that may need to be sent to the vendor. The order changes should always be referred to as 'versions' and kept clearly distinct from order 'revisions' which are vendor changes uploaded via the ediupack program.

Thread assignment program (ORDER_REVISION_THREAD) will filter eligible records from revision history (REV_ORDERS) and order header (ORDHEAD) tables based on its order status ('A' approved or 'C'losed) and supplier that exists from supplier (SUPS) table. When orders are approved or when approved orders are modified, this program selects order numbers from the REV_ORDERS table. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_ORDER_REVISION_STG.

The Business logic program (ORDER_REVISION) will process all records from the staging table. Using bulk processing, this program will writes versions of approved orders and its current order information to the order/allocation revision history tables. After the new version has been written to the order revision tables, all records will be deleted from the REV_ORDERS table for that order record. If an order is not in approved status at the time the batch program runs, then none of the above processing will occur. These records will stay on the REV_ORDERS table until the PO is approved

or deleted. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 5–7 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 6 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Order number.

Restart/Recovery

NA

Key Tables Affected

Table 5–8 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_ORDER_REVISION_STG	Yes	Yes	No	Yes
REV_ORDERS	Yes	No	No	Yes
ORDHEAD	Yes	No	Yes	No
SUPS	Yes	No	No	No
ORDHEAD_REV	Yes	Yes	No	No
ORDSKU	Yes	No	No	No
ORDLOC	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No
ORDSKU_REV	No	Yes	No	No
ORDLOC_REV	No	Yes	No	No
ALLOC_REV	No	Yes	No	No

Design Assumptions

NA

ordprg (Purge Aged Purchase Orders)

Module Name	ordprg.pc
Description	Purge Aged Purchase Orders
Functional Area	Purchase Orders
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS285
Runtime Parameters	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to remove old purchase orders from the system.

If importing is not enabled in the system (as defined by the import system indicator = 'N') and if invoice matching is not installed, then all details associated with an order are deleted when the order has been closed for more months than specified in 'Order History Months' purge parameter. Orders will only be deleted if all allocations associated, if any, have been closed.

If invoice matching is installed, then all details associated with an order are deleted when the order has been closed for more months than specified in the 'Order History Months' purge parameter. Orders are deleted only if allocations associated have been closed, shipments from the order have been completely matched to invoices or closed, and all those invoices have been posted.

If importing is enabled in the system (as defined by the import system indicator = 'Y') and if invoice matching is not installed, then all details associated with the order are deleted when the order has been closed for more months than specified in the 'Order History Months' purge option. This action presupposes that all ALC records associated with an order are in 'Processed' status, specified in ALC_HEAD (status) and allocations associated to the order, if any, have been closed.

If invoice matching is installed, then all details associated with an order are deleted when the order has been closed for more months than specified in the 'Order History Months' purge parameter. This action presupposes that all ALC records associated with an order are in 'Processed' status, specified in ALC_HEAD (status), all allocations associated to the order, if any, have been closed, all shipments from the order have been completely matched to invoices or closed, and all those invoices have been posted.

If the order to be purged is an import PO and it doesn't have a letter of credit (LC) then purge the related records related to obligations, ALC and ICB transfers.

Restart/Recovery

Restart ability will be implied, because the records that are selected from the driving cursor will be deleted before the commit. Restart library functions will still be included

to ensure that rollback segments are not exceeded (by committing at intervals) and to perform basic record keeping functionality.

Design Assumptions

N/A

order_purge_job (Purge Aged Purchase Orders)

Module Name	order_purge_job
Description	Purge Aged Purchase Orders
Functional Area	Purchase Orders
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (ORDER_PURGE_THREAD) will filter eligible records from order header (ORDHEAD) and other associated and order-related tables based on its conditions below:

1. If importing is not enabled in the system (as defined by the import system indicator = 'N') and if invoice matching is not installed, then all details associated with an order are deleted when the order has been closed for more months than specified in 'Order History Months' purge parameter. Orders will only be deleted if all allocations associated, if any, have been closed.
2. If invoice matching is installed, then all details associated with an order are deleted when the order has been closed for more months than specified in the 'Order History Months' purge parameter. Orders are deleted only if allocations associated have been closed, shipments from the order have been completely matched to invoices or closed, and all those invoices have been posted.
3. If importing is enabled in the system (as defined by the import system indicator = 'Y') and if invoice matching is not installed, then all details associated with the order are deleted when the order has been closed for more months than specified in the 'Order History Months' purge option. This action presupposes that all ALC records associated with an order are in 'Processed' status, specified in ALC_HEAD (status) and allocations associated to the order, if any, have been closed.
4. If invoice matching is installed, then all details associated with an order are deleted when the order has been closed for more months than specified in the 'Order History Months' purge parameter. This action presupposes that all ALC records associated with an order are in 'Processed' status, specified in ALC_HEAD (status), all allocations associated to the order, if any, have been closed, all shipments from the order have been completely matched to invoices or closed, and all those invoices have been posted.
5. If the order to be purged is an import PO and it doesn't have a letter of credit (LC) then purge the related records related to obligations, ALC and ICB transfers.

These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_ORDER_PURGE_STG.

The Business logic program (ORDER_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from order header (ORDHEAD) and other associated and order-related tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 5–9 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Order number

Restart/Recovery

NA

Key Tables Affected

Table 5–10 Key Tables Affected

Table	Select	Insert	Update	Delete
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_ORDER_PURGE_STG	Yes	Yes	No	Yes
ORDHEAD	Yes	No	No	Yes
ORDLC	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	Yes
SHIPMENT	Yes	No	No	Yes
SHIPSKU	Yes	No	Yes	Yes
INVC_HEAD	Yes	No	No	Yes
ORDLOC_REV	No	No	No	Yes
ORDHEAD_REV	No	No	No	Yes
ALLOC_REV	No	No	No	Yes

Table 5–10 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ALC_HEAD	Yes	No	No	Yes
ALC_COMP_LOC	No	No	No	Yes
OBLIGATION_COMP_LOC	No	No	No	Yes
OBLIGATION_COMP	No	No	No	Yes
OBLIGATION	No	No	No	Yes
TRANSPORTATION	Yes	No	No	Yes
MISSING_DOC	No	No	No	Yes
TRANS_PACKING	No	No	No	Yes
TRANS_DELIVERY	No	No	No	Yes
TRANS_CLAIMS	No	No	No	Yes
TRANS_LIC_VISA	No	No	No	Yes
TRANS_SKU	No	No	No	Yes
CE_ORD_ITEM	Yes	No	No	Yes
CE_LIC_VISA	No	No	No	Yes
CE_CHARGES	No	No	No	Yes
CE_SHIPMENT	No	No	No	Yes
CE_PROTEST	No	No	No	Yes
CE_FORMS	No	No	No	Yes
CE_HEAD	v	No	No	Yes
APPT_HEAD	Yes	No	No	Yes
APPT_DETAIL	Yes	No	No	Yes
DOC_CLOSE_QUEUE	No	No	No	Yes
DAILY_PURGE	No	Yes	No	No
ORDSKU	Yes	No	No	Yes
ITEM_MASTER	Yes	No	No	No
PACKITEM	Yes	No	No	No
PACK_TMPL_HEAD	Yes	No	No	No
RTV_DETAIL	No	No	No	Yes
WO_DETAIL	No	No	No	Yes
CARTON	No	No	No	Yes
WO_HEAD	Yes	No	No	Yes
ALLOC_CHRG	No	No	No	Yes
ALLOC_DETAIL	No	No	No	Yes
TIMELINE	No	No	No	Yes
ORDLOC	No	No	No	Yes
ORDLOC_DISCOUNT	No	No	No	Yes
ORDLOC_EXP	No	No	No	Yes

Table 5–10 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ORDSKU_HTS_ASSESS	No	No	No	Yes
ORDSKU_HTS	No	No	No	Yes
REQ_DOC	No	No	No	Yes
ORDSKU_REV	No	No	No	Yes
ORDLOC_INVC_COST	No	No	Yes	Yes
ORDCUST	Yes	No	No	Yes
ORDCUST_DETAIL	No	No	No	Yes
ORDCUST_CUSTOMER_DETAIL	No	No	No	Yes
ORD_XDOCK_TEMP	No	No	No	Yes
INVC_XREF	No	No	No	Yes
INVC_MATCH_WKSHT	No	No	No	Yes
ORDLOC_WKSHT	No	No	No	Yes
SUP_VIOLATION	No	No	No	Yes
REV_ORDERS	No	No	No	Yes
LC_ORDAPPLY	No	No	No	Yes
ORDHEAD_DISCOUNT	No	No	No	Yes
RUA_RIB_INTERFACE	No	No	No	Yes
ORDLOC_TEMP	No	No	No	Yes
ALLOC_CHRG_TEMP	No	No	No	Yes
ALLOC_DETAIL_TEMP	No	No	No	Yes
ALLOC_HEADER_TEMP	No	No	No	Yes
ORDSKU_TEMP	No	No	No	Yes
ORDLOC_EXP_TEMP	No	No	No	Yes
ORDSKU_HTS_ASSESS_TEMP	No	No	No	Yes
ORDSKU_HTS_TEMP	No	No	No	Yes
ORDLOC_DISCOUNT_TEMP	No	No	No	Yes
TIMELINE_TEMP	No	No	No	Yes
REQ_DOC_TEMP	No	No	No	Yes
WO_DETAIL_TEMP	No	No	No	Yes
WO_HEAD_TEMP	No	No	No	Yes
REPL_RESULTS_TEMP	No	No	No	Yes
DEAL_COMP_PROM	No	No	No	Yes
DEAL_HEAD	Yes	No	No	Yes
DEAL_THRESHOLD	No	No	No	Yes
DEAL_DETAIL	No	No	No	Yes

Table 5–10 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
DEAL_QUEUE	No	No	No	Yes
ORD_INV_MGMT	No	No	No	Yes
REPL_RESULTS	No	No	No	Yes
INVC_DETAIL	No	No	No	Yes
INVC_NON_MERCH	No	No	No	Yes
INVC_MERCH_VAT	No	No	No	Yes
INVC_DETAIL_VAT	No	No	No	Yes
INVC_DISCOUNT	No	No	No	Yes
INVC_TOLERANCE	No	No	No	Yes
INVC_MATCH_QUEUE	No	No	No	Yes
TSFHEAD	No	No	No	Yes
TSFDETAIL	No	No	No	Yes
TSFDETAIL_CHRG	No	No	No	Yes
DEAL_ITEMLOC_ITEM	No	No	No	Yes
DEAL_ITEMLOC_DCS	No	No	No	Yes
DEAL_ITEMLOC_DIV_GRP	No	No	No	Yes
DEAL_ITEMLOC_PARENT_DIFF	No	No	No	Yes
ORDHEAD_L10N_EXT	No	No	No	Yes
ORD_TAX_BREAKUP	No	No	No	Yes
ORDHEAD_CFA_EXT	No	No	No	Yes
DEALHEAD_CFA_EXT	No	No	No	Yes
TSFHEAD_CFA_EXT	No	No	No	Yes
SHIPSKU_LOC_PRG_HIST	No	Yes	No	No
SHIPSKU_PRG_HIST	No	Yes	No	No
SHIPMENT_PRG_HIST	No	Yes	No	No
ALLOC_CHRG_PRG_HIST	No	Yes	No	No
ALLOC_DETAIL_PRG_HIST	No	Yes	No	No
ALLOC_HEADER_PRG_HIST	No	Yes	No	No
ORDLOC_REV_PRG_HIST	No	Yes	No	No
ORDSKU_REV_PRG_HIST	No	Yes	No	No
ORDHEAD_REV_PRG_HIST	No	Yes	No	No
ORDCUST_DETAIL_PRG_HIST	No	Yes	No	No
ORDCUST_PRG_HIST	No	Yes	No	No

Table 5–10 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ORDLOC_CFA_EXT_PRG_HIST	No	Yes	No	No
ORDLOC_PRG_HIST	No	Yes	No	No
ORDLOC_DISCOUNT_PRG_HIST	No	Yes	No	No
ORDLOC_EXP_PRG_HIST	No	Yes	No	No
ORDSKU_HTS_ASSESS_PRG_HIST	No	Yes	No	No
ORDSKU_HTS_PRG_HIST	No	Yes	No	No
ORDSKU_CFA_EXT_PRG_HIST	No	Yes	No	No
ORDSKU_PRG_HIST	No	Yes	No	No
ORDHEAD_DISCOUNT_PRG_HIST	No	Yes	No	No
ORDHEAD_L10N_EXT_PRG_HIST	No	Yes	No	No
ORD_TAX_BREAKUP_PRG_HIST	No	Yes	No	No
ORDHEAD_CFA_EXT_PRG_HIST	No	Yes	No	No
ORDHEAD_PRG_HIST	No	Yes	No	No

Design Assumptions

NA

poindbatch.ksh (Upload Order Data)

Module Name	poindbatch.ksh
Description	Upload Order Data
Functional Area	Purchase Order Maintenance
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS234
Wrapper Script	rmswrap_shell_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to Bulk upload xml file data from template files to S9T_FOLDER table (into content_xml column).

This batch will be responsible for validating the input parameters, below are the list of validations.

- The Input file should exist.
- The Input file's extension must be ".xml".
- The template_name should be valid. Function S9T_PKG.CHECK_TEMPLATE is called for validation.
- Destination (Optional Parameter) should be STG or Merchandising. If destination is not passed then default it to STG.

Once xml data is loaded into S9T_FOLDER table, the script will do post processing by calling the packages listed below:

- PO_INDUCT_SQL.INIT_PROCESS - This initialize a row in svc_process_tracker for asynchronous processing.
- PO_INDUCT_SQL.EXEC_ASYNC - This function calls the main induction process that uploads data into the staging tables, validates and inserts data into the base Merchandising purchase order tables.

Restart/Recovery

N/A

Design Assumptions

N/A

po_indctn_purge.ksh (Purge PO Induction Staging Tables)

Module Name	po_indctn_purge.ksh
Description	Purge PO induction staging tables
Functional Area	Purchase Orders
Module Type	Admin
Module Technology	Shell Script
Catalog ID	RMS499
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to remove old purchase order records from the staging tables. Records that are candidates for deletion are:

- Processes that have successfully been processed or processed with warnings that have been uploaded to Merchandising or downloaded to S9T
- Processes that have status = 'PE' processed with errors and have no liked data

- Processes in error status where all other related records containing the process ID have been processed successfully
- Processes that are passed the data retention days (system_options.proc_data_retention_days)
- All order records within a process where all related records for the order in the other staging tables are successfully uploaded to Merchandising. The process tracker record should not be deleted if there are other orders that are not uploaded to Merchandising.

Restart/Recovery

Restart ability will be implied, because the records that are selected from the cursor will be deleted before the commit.

Design Assumptions

N/A

6

Deals

Deals are complex business processes that can either affect the cost a retailer pays for goods purchased from a supplier (off invoice deals) or generate income from suppliers/partners (billback/rebate deals). These basic types of deals require different processing. This chapter contains information about the batch processes that support all types of Deals.

For additional information about Deals, including detailed flow diagrams, see the Merchandising Functional Library (Doc ID: 1585843.1).

Note: The White Papers in this library are intended only for reference and educational purposes and may not reflect the latest version of Oracle Retail software.

Program Summary

Table 6–1 Deals - Program Summary

Program	Description
dealupld.pc	Upload of Deals from 3rd Party Systems
batch_ditinsrt.ksh	Deal Calculation Queue Insert Multithreading
ditinsrt.pc	Insert into Deal Calculation Queue
discotbapply.pc	Update OTB After Deal Discounts
dealact.pc	Calculate Actual Impact of Billback Deals
dealinc.pc	Calculate Weekly/Monthly Income Based on Turnover
dealday.pc	Daily Posting of Deal Income to Stock & General Ledgers
dealfct.pc	Calculates/Update Forecasted Values for Deals
vendinvc.pc	Stage Complex Deal Invoice Information
vendinvf.pc	Stage Fixed Deal Invoice Information
dealcls.pc	Close Expired Deals
deal_close_job	Close Expired Deals
dealprg.pc	Purge Closed Deals
deal_purge_job	Purge Closed Deals
deal_actuals_purge_job	Purge Closed Deals Actuals Item/Location

dealupld (Upload of Deals from 3rd Party Systems)

Module Name	dealupld.pc
Description	Upload of Deals from 3rd Party Systems
Functional Area	Deals
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS42
Wrapper Script	rmswrap_multi_in_rej.ksh

Design Overview

Dealupld.pc uploads deals from external systems into Merchandising. Generally, deals are uploaded from merchandise suppliers and other trading partners. Dealupld uses a proprietary file format (not any EDI standard).

Both deals uploaded via dealupld.pc and deals created via the user interface are written to a series of deals tables for deals processing.

Restart/Recovery

The program uses File based restart recovery process.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000008

Integration File Layout

Table 6–2 dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type (the beginning of the input file)
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	File Type Definition	Char(5)	EDIDU	Identifies file as 'EDI Deals Upload'
	File Create Date	Char(14)	Create date	Current date, formatted to 'YYYYMMDDHH24MISS'

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
THEAD	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type to upload a new deal header
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Detail Record Type	Char(5)	DHDTL	Identifies file record type Deal Header. This record MUST BE FOLLOWED BY ONE AND ONLY ONE REQUIRED TDETL RECORD that holds the deal head information

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
TDETL	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type to upload a new deal
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Partner Type	Char(6)	REQUIRED	Type of the partner the deal applies to. Valid values are 'S' for a supplier, 'S1' for supplier hierarchy level 1 (for example, the manufacturer), 'S2' for supplier hierarchy level 2 (for example, the distributor) and 'S3' for supplier hierarchy level 3 (that is, the wholesaler). Descriptions of these codes will be held on the codes table under a code_type of 'SUHL' Information pertaining to a single deal has to belong to the same supplier, since a deal may have only one supplier hierarchy associated with it. Only items with the same supplier hierarchy can be on the same deal. Supplier hierarchy is stored at an item / supplier / country / location level
	Partner Id	Char(10)	Blank (space character string)	Level of supplier hierarchy (for example, manufacturer, distributor or wholesaler), set up as a partner in the PARTNER table, used for assigning rebates by a level other than supplier. Rebates at this level will include all eligible supplier/item/country records assigned to this supplier hierarchy level This field is required if the Partner Type field was set to 'S1', 'S2' or 'S3'. This field must be blank if the Partner Type field was set to 'S'

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Supplier	Number (10)	Blank (space character string)	Deal supplier's number. This supplier can be at any level of supplier hierarchy This field is required if the Partner Type field was set to 'S'. This field must be blank if the Partner Type field was set to 'S1', 'S2' or 'S3'
	Type	Char(6)	REQUIRED	Type of the deal. Valid values are A for annual deal, P for promotional deal, O for PO-specific deal or M for vendor-funded markdown. Deal types will be held on the codes table under a code type of 'DLHT'
	Currency Code	Char(3)	Blank (space character string)	Currency code of the deal's currency. All costs on the deal will be held in this currency If Type is 'O', 'P' or 'A', then Currency Code may not be blank. Currency Code has to be blank if Type is 'M'
	Active Date	Char(14)	REQUIRED	Date the deal will become active. This date will determine when deal components begin to be factored into item costs. For a PO-specific deal, the active_date will be the order's written date
	Close Date	Char(14)	Blank (space character string)	Date the deal will/did end. This date determines when deal components are no longer factored into item costs. It is optional for annual deals, required for promotional deals. It will be left NULL for PO-specific deals Close Date must not be blank if Type is 'P' or 'M'. Close Date has to be blank if Type is 'O'
	External Reference Number	Char(30)	Blank (space character string)	Any given external reference number that is associated with the deal
	Order Number	Number (12)	Blank (space character string)	Order the deal applies to, if the deal is PO-specific

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Recalculate Approved Orders	Char(1)	REQUIRED	Indicates if approved orders should be recalculated based on this deal once the deal is approved. Valid values are Y for yes or N for no Valid values are 'Y' and 'N'
	Comments	Char (2000)	Blank (space character string)	Free-form comments entered with the deal
	Billing Type	Char(6)	REQUIRED	Billing type of the deal component. Valid values are 'OI' for off-invoice, 'BB' for bill-back, 'VFP' for vendor funded promotion and 'VFM' for vendor funded markdown. Billing types will be held on the codes table under a code type of 'DLBT'
	Bill Back Period	Char(6)	Blank (space character string)	Code that identifies the bill-back period for the deal component. This field will only be populated for billing types of 'BB' or 'VFP' or 'VFM'. Valid bill back period codes are 'W', 'M', 'Q', 'H', 'A'. If Billing Type is 'BB' then Bill Back Period must not be blank; if Billing Type is 'OI' (off invoice), then Bill back Period has to be blank
	Deal Application Timing	Char(6)	Blank (space character string)	Indicates when the deal component should be applied - at PO approval or time of receiving. Valid values are 'O' for PO approval, 'R' for receiving. These values will be held on the codes tables under a code type of 'AALC'. It must be NULL for an M-type deal (vendor funded markdown)

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Threshold Limit Type	Char(6)	Blank (space character string	Identifies whether thresholds will be set up as qty values, currency amount values or percentages (growth rebates only). Valid values are 'Q' for qty, 'A' for currency amount. Threshold limit types will be held on the codes table under a code type of 'DLLT'. It must be NULL for an M-type deal (vendor funded markdown) or if the threshold value type is 'Q' (buy/get deals). If Growth Rebate Indicator is 'Y', then the Threshold Limit Type has to be 'Q', 'A' or NULL
	Type	Char(6)	REQUIRED	Type of the deal. Valid values are A for annual deal, P for promotional deal, O for PO-specific deal or M for vendor-funded markdown. Deal types will be held on the codes table under a code type of 'DLHT'
	Currency Code	Char(3)	Blank (space character string)	Currency code of the deal's currency. All costs on the deal will be held in this currency If Type is 'O', 'P' or 'A', then Currency Code may not be blank. Currency Code has to be blank if Type is 'M'
	Active Date	Char(14)	REQUIRED	Date the deal will become active. This date will determine when deal components begin to be factored into item costs. For a PO-specific deal, the active_date will be the order's written date
	Close Date	Char(14)	Blank (space character string)	Date the deal will/did end. This date determines when deal components are no longer factored into item costs. It is optional for annual deals, required for promotional deals. It will be left NULL for PO-specific deals Close Date must not be blank if Type is 'P' or 'M'. Close Date has to be blank if Type is 'O'

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	External Reference Number	Char(30)	Blank (space character string)	Any given external reference number that is associated with the deal
	Order Number	Number (12)	Blank (space character string)	Order the deal applies to, if the deal is PO-specific
	Recalculate Approved Orders	Char(1)	REQUIRED	Indicates if approved orders should be recalculated based on this deal once the deal is approved. Valid values are Y for yes or N for no Valid values are 'Y' and 'N'
	Comments	Char (2000)	Blank (space character string)	Free-form comments entered with the deal
	Billing Type	Char(6)	REQUIRED	Billing type of the deal component. Valid values are 'OI' for off-invoice, 'BB' for bill-back, 'VFP' for vendor funded promotion and 'VFM' for vendor funded markdown. Billing types will be held on the codes table under a code type of 'DLBT'
	Bill Back Period	Char(6)	Blank (space character string)	Code that identifies the bill-back period for the deal component. This field will only be populated for billing types of 'BB' or 'VFP' or 'VFM'. Valid bill back period codes are 'W', 'M', 'Q', 'H', 'A'. If Billing Type is 'BB' then Bill Back Period must not be blank; if Billing Type is 'OI' (off invoice), then Bill back Period has to be blank
	Deal Application Timing	Char(6)	Blank (space character string)	Indicates when the deal component should be applied - at PO approval or time of receiving. Valid values are 'O' for PO approval, 'R' for receiving. These values will be held on the codes tables under a code type of 'AALC'. It must be NULL for an M-type deal (vendor funded markdown)

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Threshold Limit Type	Char(6)	Blank (space character string)	Identifies whether thresholds will be set up as qty values, currency amount values or percentages (growth rebates only). Valid values are 'Q' for qty, 'A' for currency amount. Threshold limit types will be held on the codes table under a code type of 'DLLT'. It must be NULL for an M-type deal (vendor funded markdown) or if the threshold value type is 'Q' (buy/get deals). If Growth Rebate Indicator is 'Y', then the Threshold Limit Type has to be 'Q', 'A' or NULL
	Threshold Limit Unit of Measure	Char(4)	Blank (space character string)	Unit of measure of the threshold limits, if the limit type is quantity. Only Unit of Measures with a UOM class of 'VOL' (volume), 'MASS' or 'QTY' (quantity) can be used in this field. Valid Unit of Measures can be found on the UOM_ CLASS table If the Threshold Limit Type is 'A', then Threshold Limit Unit of Measure has to be blank. If the Threshold Limit Type is 'Q', Threshold Limit Unit of Measure must not be blank. If Threshold Limit Type is blank, Threshold Limit Unit of Measure must be blank.
	Rebate Indicator	Char(1)	REQUIRED	Indicates if the deal component is a rebate. Deal components can only be rebates for bill-back billing types. Valid values are 'Y' for yes or 'N' for no. If Billing Type is 'OI', then Rebate Indicator must be 'N'.

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Rebate Calculation Type	Char(6)	Blank (space character string)	Indicates if the rebate should be calculated using linear or scalar calculation methods. Valid values are 'L' for linear or 'S' for scalar. This field will be required if the rebate indicator is 'Y'. Rebate calculation types will be held on the codes table under a code type of 'DLCT' If Rebate Indicator is 'Y', then Rebate Calculation Type must not be blank. Otherwise it has to be blank.
	Growth Rebate Indicator	Char(1)	REQUIRED	Indicates if the rebate is a growth rebate, meaning it is calculated and applied based on an increase in purchases or sales over a specified period of time. Valid values are 'Y' for yes or 'N' for no If Rebate Indicator is 'N', then Growth Rebate Indicator must be 'N'.
	Historical Comparison Start Date	Char(14)	Blank (space character string)	The first date of the historical period against which growth will be measured in this growth rebate. Note performance and the rebate amount are not calculated - this field is for informational/reporting purposes only If Growth Rebate Indicator is 'Y', then Historical Comparison Start Date must not be blank. Otherwise it must be blank.
	Historical Comparison End Date	Char(14)	Blank (space character string)	The last date of the historical period against which growth will be measured in this growth rebate. Note performance and the rebate amount are not calculated - this field is for informational/reporting purposes only If Growth Rebate Indicator is 'Y', then Historical Comparison End Date must not be blank. Otherwise it must be blank.

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Rebate Purchases or Sales Application Indicator	Char(6)	Blank (space character string)	Indicates if the rebate should be applied to purchases or sales. Valid values are 'P' for purchases or 'S' for sales. It will be required if the rebate indicator is 'Y'. Rebate purchase/sales indicators will be held on the codes table under a code type of 'DLRP' If the Rebate Indicator is 'Y', then the Rebate Purchases or Sales Application Indicator must not be blank. Otherwise it has to be blank.
	Security Indicator	Char	Y	Security Indicator
TTAIL	File Line Identifier	Char(5)	TTAIL	Identifies file record type (the end of the transaction detail)
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Record Counter	Numeric ID(6)	Sequential number Created by program.	Number of records/transactions in current transaction set (only records between thead and ttail). For DHDTL TDETL records this will always be 1
THEAD	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type to upload a new deal sub loop
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Detail Record Type	Char(5)	DCDTL	Identifies file record type of sub loop as Deal Component Detail
TDETL	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type to upload deal components
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Deal Component Type	Char(6)	REQUIRED	Type of the deal component, user-defined and stored on the DEAL_COMP_TYPE table

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Application Order	Number (10)	Blank (space character string)	Number indicating the order in which the deal component should be applied with respect to any other deal components applicable to the item within the deal. This number will be unique across all deal components within the deal. It must be NULL for an M-type deal (vendor funded markdown)
	Collect Start Date	Char(14)	Blank (space character Deal Component Type string)	Date that collection of the bill-back should begin If Billing Type is 'BB' then Collect Start Date must not be blank, otherwise it has to be blank
	Collect End Date	Char(14)	Blank (space character string)	Date that collection of the bill-back should end If Billing Type is 'BB' then Collect End Date must not be blank, otherwise it has to be blank
	Cost Application Level Indicator	Char(6)	Blank (space character string)	Indicates what cost bucket the deal component should affect. Valid values are 'N' for net cost, 'NN' for net cost and 'DNN' for dead net cost. These values will be held on the codes tables under a code type of 'DLCA'. It must be NULL for an M-type deal (vendor funded markdown)
	Pricing Cost Indicator	Char(1)	REQUIRED	Identifies deal components that should be included when calculating a pricing cost Valid values are 'Y'es and 'N'o

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Deal Class	Char(6)	Blank (space character string)	Identifies the calculation class of the deal component. Valid values are 'CU' for cumulative (discounts are added together and taken off as one lump sum), 'CS' for cascade (discounts are taken one at a time with subsequent discounts taken off the result of the previous discount) and 'EX' for exclusive (overrides all other discounts). 'EX' type deal components are only valid for promotional deals. Deal classes will be held on the codes table under a code type of 'DLCL'. It must be NULL for an M-type deal (vendor funded markdown)
	Threshold Value Type	Char(6)	Blank (space character string)	Identifies whether the discount values associated with the thresholds will be set up as qty values, currency amount values, percentages or fixed amounts. Valid values are 'Q' for qty, 'A' for currency amount, 'P' for percentage or 'F' for fixed amount. Qty threshold value (buy/get) deals are only allowed on off-invoice discounts. Deal threshold value types will be held on the codes table under a code type of 'DLL2'. It must be NULL for an M-type deal (vendor funded markdown). If Billing Type is 'BB', then the Threshold Value Type must be'A' or 'P'
	Buy Item	Char(25)	Blank (space character string)	Identifies the item that must be purchased for a quantity threshold-type discount. This value is required for quantity threshold value type discounts. Otherwise it has to be blank

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Get Type	Char(6)	Blank (space character string)	Identifies the type of the 'get' discount for a quantity threshold-type (buy/get) discount. Valid values include 'X' (free), 'P' (percent), 'A' (amount) and 'F' (fixed amount). They are held on the codes table under a code type of 'DQGT'. This value is required for quantity threshold value deals. Otherwise it has to be blank
	Get Value	Number(20,4)	All 0s.	Identifies the value of the 'get' discount for a quantity threshold-type (buy/get) discount that is not a 'free goods' deal. The Get Type above identifies the type of this value. This value is required for quantity threshold value type deals that are not a Get Type of free. Otherwise it has to be 0 If Get Type is ‘P’, ‘A’ or ‘F’, then Get Value must not be blank. If the Get Type is ‘X’ or blank, then Get Value has to be blank
	Buy Item Quantity	Number(12,4)	All 0s.	Identifies the quantity of the threshold 'buy' item that must be ordered to qualify for the 'free' item. This value is required for quantity threshold value type discounts. Otherwise it has to be 0
	Recursive Indicator	Char(1)	REQUIRED	For 'buy/get free' discounts, indicates if the quantity threshold discount is only for the first 'buy amt.' purchased (such as,. for the first 10 purchased, get 1 free), or if a free item will be given for every multiple of the 'buy amt' purchased on the order (such as,. for each 10 purchased, get 1 free). Valid values are 'Y' for yes or 'N' for no If the Get Type is blank, then Recursive Indicator has to be ‘N’

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Buy Item Order Target Quantity	Number(12,4)	All 0s.	Indicates the targeted purchase level for all locations on a purchase order. This is the target level that will be used for future calculation of net cost. This value is required for quantity threshold value type deals. Otherwise it has to be 0
	Average Buy Item Order Target Quantity Per Location	Number(12,4)	All 0s.	Indicates the average targeted purchase level per location on the deal. This value will be used in future cost calculations. This value is required for quantity threshold value type deals. Otherwise it has to be 0
	Get Item	Char(25)	Blank (space character string)	Identifies the 'get' item for a quantity threshold-type (buy/get) discount. This value is required for quantity threshold value deals. Otherwise it has to be blank If Get Type is 'P', 'A', 'F' or 'X', then Get Item must not be blank. If the Get Type is blank, then Get Item has to be blank
	Get Quantity	Number(12,4)	All 0s.	Identifies the quantity of the identified 'get' item that will be given at the specified 'get' discount if the 'buy amt' of the buy item is purchased. This value is required for quantity threshold value type discounts. Otherwise it has to be 0 If Get Type is 'P', 'A', 'F' or 'X', then Get Quantity must not be 0. If the Get Type is blank, then Get Quantity has to be 0

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Free Item Unit Cost	Number(20,4)	All 0s.	For 'buy/get free' discounts, identifies the unit cost of the threshold 'free' item that will be used in calculating the prorated qty. discount. It will default to the item/supplier cost, but can be modified based on the agreement with the supplier. It must be greater than zero as this is the cost that would normally be charged for the goods if no deal applied If Get Type is 'P', 'A', 'F' or blank, then Free Item Unit Cost must be 0. If the Get Type is 'X', then Free Item Unit Cost must not be 0
	Transaction Level Discount Indicator	Char(1)	REQUIRED	Indicates if the discount is a transaction-level discount (such as,. 10% across an entire PO) Valid Values are 'Y' or 'N'. If set to 'Y', Deal Class has to be 'CU' and Billing Type has to be 'OI'. No DIDTL or PPDTL records may be present for a Transaction Level Discount DCDTL record
	Comments	Char(2000)	Blank (space character string)	Free-form comments entered with the deal component
TTAIL	File Line Identifier	Char(5)	TTAIL	Identifies file record type (the end of the transaction detail)
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Record Counter	Numeric ID(6)	Sequential number Created by program.	Number of records/transactions in current transaction set (only records between thead and ttail)

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
THEAD	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type to upload a new deal sub loop
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Detail Record Type	Char(5)	DIDTL	Identifies file record type of sub loop as Deal Component Item-location Detail

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
TDETL	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type to upload deal item-location details
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Merchandise Level	Char(6)	REQUIRED	Indicates what level of the merchandise hierarchy the record is at. Valid values include '1' for company-wide (all items), '2' for division, '3' for group, '4' for dept, '5' for class, '6' for subclass, '7' for line, '8' for line/differentiator 1, '9' for line/differentiator 2' '10' for line/differentiator 3, '11' for line/differentiator 4 and '12' for. These level types will be held on the codes table under a code type of 'DIML'
	Company Indicator	Char(1)	REQUIRED	Indicates if the deal component is applied company-wide (that is, whether all items in the system will be included in the discount or rebate). Valid values are 'Y' for yes and 'N' for no
	Division	Number (4)	Blank (space character string)	ID of the division included in or excluded from the deal component. Valid values are on the DIVISION table If Group is not blank, then Division must not be blank. If Merchandise Level is 2, then Division must not be blank and Group, Department, Class and Subclass must be blank
	Group	Number (4)	Blank (space character string).	ID of the group included in or excluded from the deal component. Valid values are on the GROUPS table If Department is not blank, then Group must not be blank. If Merchandise Level is 3, then Group must not be blank and Department, Class and Subclass must be blank

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Department	Number (4)	Blank (space character string).	ID of the department included in or excluded from the deal component. Valid values are on the DEPS table If Class is not blank, then Department must not be blank. If Merchandise Level is 4, then Department must not be blank and Class and Subclass must be blank
	Class	Number (4)	Blank (space character string).	ID of the class included in or excluded from the deal component. Valid values are on the CLASS table If Subclass is not blank, then Class must not be blank. If Merchandise Level is 5, then Class must not be blank and Subclass must be blank
	Subclass	Number (4)	Blank (space character string).	ID of the subclass included in or excluded from the deal component. Valid values are on the SUBCLASS table If Merchandise Level is 6 or more than 6, then Subclass must not be blank
	Item Parent	Char(25)	Blank (space character string)	Alphanumeric value that uniquely identifies the item/group at the level above the item. This value must exist as an item in another row on the ITEM_MASTER table If Merchandise Level is 7, then Item Parent or Item Grandparent must not be blank (at least one of them has to be given)
	Item Grandparent	Char(25)	Blank (space character string)	Alphanumeric value that uniquely identifies the item/group two levels above the item. This value must exist as both an item and an item parent in another row on the ITEM_MASTER table If Merchandise Level is 7, then Item Parent or Item Grandparent must not be blank (at least one of them has to be given)

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Differentiator 1	Char(10)	Blank (space character string)	Diff_group or diff_id that differentiates the current item from its item_parent If Item Grandparent, Item Parent and Differentiator 2 are blank, then Differentiator 1 must be blank. If Merchandise Level is 8, then Differentiator 1 must not be blank
	Differentiator 2	Char(10)	Blank (space character string)	Diff_group or diff_id that differentiates the current item from its item_parent If Item Grandparent, Item Parent and Differentiator 1 are blank, then Differentiator 2 must be blank. If Merchandise Level is 9, then Differentiator 2 must not be blank
	Differentiator 3	Char(10)	Blank (space character string)	Diff_group or diff_id that differentiates the current item from its item_parent If Item Grandparent, Item Parent and Differentiator 1 and 2 are blank, then Differentiator 3 must be blank. If Merchandise Level is 10, then Differentiator 3 must not be blank
	Differentiator 4	Char(10)	Blank (space character string)	Diff_group or diff_id that differentiates the current item from its item_parent If Item Grandparent, Item Parent and Differentiator 1, 2 and 3 are blank, then Differentiator 4 must be blank. If Merchandise Level is 10, then Differentiator 4 must not be blank
	Organizational Level	Char(6)	Blank (space character string)	Indicates what level of the organizational hierarchy the record is at. Valid values include '1' for chain, '2' for area, '3' for region, '4' for district and '5' for location. These level types will be held on the codes table under a code type of 'DIOL' If company indicator is N, this must not be blank. If location type is warehouse or location list, this must be 5

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Chain	Number (10)	Blank (space character string).	ID of the chain included in or excluded from the deal component. Valid values are on the CHAIN table If org. level is 1, this field must not be blank
	Area	Number (10)	Blank (space character string).	ID of the area included in or excluded from the deal component. Valid values are on the AREA table If org. level is 2, this field and chain must not be blank
	Region	Number (10)	Blank (space character string).	ID of the region included in or excluded from the deal component. Valid values are on the REGION table If org. level is 3, this field, area, and chain must not be blank
	District	Number (10)	Blank (space character string).	ID of the district included in or excluded from the deal component. Valid values are on the DISTRICT table If org. level is 4, then this field, region, area, and chain must not be blank
	Location	Number (10)	Blank (space character string).	ID of the location included in or excluded from the deal component. Valid values are on the STORE, WH, or LOC_LIST_HEAD table If org. level is 5, this field must not be blank. Chain, area, region, and district should be blank if the loc_ type is L or W. If the loc_ type is S, then they all must not be blank If Location Type is not blank, then Location must not be blank. Otherwise it has to be blank
	Origin Country Identifier	Char(3)	Blank (space character string)	Origin country of the item that the deal component should apply to
	Location Type	Char(1)	Blank (space character string)	Type of the location referenced in the location field. Valid values are 'S' and 'W'. Location types will be held on the codes table under the code type 'LOC3' If location is blank then this field has to be blank also

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Item	Char(25)	Blank (space character string)	Unique alphanumeric value that identifies the item If Merchandise Level is 10, then Item must not be blank
	Exclusion Indicator	Char(1)	REQUIRED	Indicates if the deal component item/location line is included in the deal component or excluded from it. Valid values are 'Y' for yes or 'N' for no
	Reference Line	Number (10)	REQUIRED	This value determines which line in the input file this item-loc record belongs to
TTAIL	File Line Identifier	Char(5)	TTAIL	Identifies file record type (the end of the transaction detail)
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Record Counter	Numeric ID(6)	Sequential number Created by program.	Number of records/transactions in current transaction set (only records between thead and ttail)
THEAD	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type to upload a new deal sub loop
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Detail Record Type	Char(5)	PPDTL	Identifies file record type of sub loop as Proof of Performance Detail
TDETL	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type to upload deal proof of performance details
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Deal Sub Item	Char(25)	No data	Specific transaction level (or below) item that's proof of performance is being measured. This can be populated when the deal itself is on a case UPC but the proof of performance is on an individual selling unit

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Proof of Performance Type	Char(6)	REQUIRED	Code that identifies the proof of performance type (that is, the term is that the item must be displayed on an end cap for 28 days - the pop_type is code 'ECD' for end cap display). Valid values for this field are stored in the code_type = 'PPT'. This field is required by the database
	Proof of Performance Value	Number (20,4)	All 0s.	Value that describes the term of the proof of performance type (that is, the term is that the item must be displayed on an end cap for 28 days - the pop_value is 28). This field is required by the database if the record has a pop_ value_type If Proof of Performance Value is not blank, then Proof of Performance Value Type must not be blank. If Proof of Performance Value is blank, then Proof of Performance Value Type must be blank
	Proof of Performance Value Type	Char(6)	Blank (space character string)	Value that describes the type of the pop_value (that is, the term is that the item must be displayed on an end cap for 28 days - the pop_value_type is the code 'DAYS' for days). Valid values for this field are stored in the code_type = 'PPVT'. This field is required by the database if the record has a pop_value If Proof of Performance Value is not blank, then Proof of Performance Value Type must not be blank. If Proof of Performance Value is blank, then Proof of Performance Value Type must be blank
	Vendor Recommend ed Start Date	Char(14)	Blank (space character string)	This column holds the date that the vendor recommends that the POP begin
	Vendor Recommend ed End Date	Char(14)	Blank (space character string)	This column holds the date that the vendor recommends that the POP end

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Planned Start Date	Char(14)	Blank (space character string)	This column holds the date that the merchandiser/category manager plans to begin the POP
	Planned End Date	Char(14)	Blank (space character string)	This column holds the date that the merchandiser/category manager plans to end the POP
	Comment	Comment Char(255)	Blank (space character string)	Free-form comments
	Reference Line	Number (10)	REQUIRED	This value determines which line in the input file this Proof of Performance record belongs to

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
TTAIL	File Line Identifier	Char(5)	TTAIL	Identifies file record type (the end of the transaction detail)
	File Line Identifier	Numeric ID(10)	Sequential number Created by program	ID of current line being read from input file
	Transaction Record Counter	Numeric ID(6)	Sequential number Created by program.	Number of records/transactions in current transaction set (only records between thead and ttail)
THEAD	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type to upload a new deal sub loop
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Detail Record Type	Char(5)	DTDTL	Identifies file record type of sub loop as Deal Component Threshold Detail

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
TDETL	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type to upload deal threshold details
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Lower Limit	Number (20,4)	REQUIRED	Lower limit of the deal component. This is the minimum value that must be met in order to get the specified discount. This value will be either a currency amount or quantity value, depending on the value in the deal_detail.threshold_limit_type field of this deal component (Threshold Value Type field of the DCDTL record that this DTDTL record belongs to as specified in the reference line field)
	Upper Limit	Number (20,4)	REQUIRED	Upper limit of the deal component. This is the maximum value for which the specified discount will apply. This value will be either a currency amount or quantity value, depending on the value in the deal_detail.threshold_limit_type field of this deal component (Threshold Value Type field of the DCDTL record that this DTDTL record belongs to as specified in the reference line field)
	Value	Number (20,4)	REQUIRED	Value of the discount that will be given for meeting the specified thresholds for this deal component. This value will be either a currency amount or quantity value, depending on the value in the deal_detail.threshold_value_type field of this deal component (Threshold Value Type field of the DCDTL record that this DTDTL record belongs to as specified in the reference line field)

Table 6–2 (Cont.) dealupld.pc - Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Target Level Indicator	Char(1)	REQUIRED	Indicates if a threshold level is the targeted purchase or sales level for a deal component. This indicator will be used for cost calculations. Valid values are 'Y' for yes and 'N' for no
	Reference Line	Number (10)	REQUIRED	This value determines which line in the input file this Threshold record belongs to
TTAIL	File Line Identifier	Char(5)	TTAIL	Identifies file record type (the end of the transaction detail)
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	Transaction Record Counter	Numeric ID(6)	Sequential number Created by program.	Number of records/transactions in current transaction set (only records between thead and ttail)
FTAIL	File Line Identifier	Char(5)	FTAIL	Identifies file record type (the end of the input file)
	File Line Identifier	Numeric ID(10)	Sequential number Created by program.	ID of current line being read from input file
	File Record Counter	Numeric ID(10)	Sequential number Created by program.	Number of records/transactions in current file (only records between head and tail)

The input file structure should be as below:

```
FHEAD
{
THEAD of DHDTL    REQUIRED   for deal head record
   TDETL          REQUIRED   1 deal head record
   TTAIL          REQUIRED   end of deal head record
   THEAD of DCDTL REQUIRED   for deal component records
   [
      TDETL       OPTIONAL   for deal component records
   ]
   TTAIL          REQUIRED   end of deal component records
   THEAD of DIDTL REQUIRED   for item-loc records
   [
      TDETL       OPTIONAL   for item-loc records
   ]
   TTAIL          REQUIRED   end of item-loc records
   THEAD of PPDTL REQUIRED   for proof of performance records
   [
```

```
      TDETL       OPTIONAL   for proof of performance records
   ]
   TTAIL          REQUIRED   end of proof of performance records
   THEAD of DTDTL REQUIRED   for threshold records
   [
      TDETL       OPTIONAL   for threshold records
   ]
   TTAIL          REQUIRED   end of threshold records
}
FTAIL
```

Design Assumptions

N/A

batch_ditinsrt.ksh (Deal Calculation Queue Insert Multithreading)

Module Name	batch_ditinsrt.ksh
Description	Deal Calculation Queue Insert Multithreading
Functional Area	Deals
Module Type	Business Processing
Module Technology	Ksh
Catalog ID	RMS187
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to multithread the ditinsrt batch program.

Restart/Recovery

A commit occurs when all details of a deal are processed. Inherent restart/recovery is achieved through deleting deals from the DEAL_QUEUE table when they are processed. Because DEAL_QUEUE is part of the driving cursor, processed deals will not be fetched again when the program restarts.

Design Assumptions

N/A

ditinsrt (Insert into Deal Calculation Queue)

Module Name	ditinsrt.pc
Description	Insert into Deal Calculation Queue
Functional Area	Deals

Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS217
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program will populate the DEAL_CALC_QUEUE table with orders that may be affected by non vendor-funded, non PO-specific deals that are on the DEAL_QUEUE table (for future processing by orddscnt.pc).

Orders that had been applied to deals that no longer apply will also be inserted into the DEAL_CALC_QUEUE table. Processed records will then be deleted from the DEAL_QUEUE table

Restart/Recovery

A commit occurs when all details of a deal are processed.

Inherent restart/recovery is achieved through deleting deals from the DEAL_QUEUE table when they are processed. Because DEAL_QUEUE is part of the driving cursor, processed deals will not be fetched again when the program restarts.

Design Assumptions

N/A

discotbapply (Update OTB After Deal Discounts)

Module Name	discotbapply.pc
Description	Update OTB After Deal Discounts
Functional Area	Deals
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS215
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Deals processing can change the cost on purchase orders. When this occurs (in the batch program orddscnt.pc), Open To Buy (OTB) must also be updated to ensure that budgets reflect reality. This program updates these OTB buckets.

Restart/Recovery

This program has inherent restart ability, because records are deleted from DISC_OTB_APPLY as they are processed. Array processing is used. Records are array fetched from DISC_OTB_APPLY table, processed and committed to the database.

Schedule

Oracle Retail Merchandising Batch Schedule

dealact (Calculate Actual Impact of Billback Deals)

Module Name	dealact.pc
Description	UCalculate Actual Impact of Billback Deals
Functional Area	Deals
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS206
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will run on a daily basis and calculate actuals information to update the deal actuals table at the item/location level for bill back non rebate deals, bill back purchase order rebate deals and bill back sales and receipts deals.

Restart/Recovery

The database commit will take place when the number of deal_id/deal_detail_id records processed is equal to commit max counter in the restart control table.

Design Assumptions

N/A

dealinc (Calculate Weekly/Monthly Income Based on Turnover)

Module Name	dealinc.pc
Description	Calculate Weekly/Monthly Income Based on Turnover

Functional Area	Deals
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS211
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program generates income for each item/location for bill-back deals.

Dealinc.pc retrieves deal attributes and actuals data from the deals tables for complex deals. It then calculates the income and will update the actuals table with the calculated income value. Additionally the program will insert the income value into the TEMP_TRAN_DATA table using the tran types deal sales and deal purchases.

Subsequent programs will run to perform forecast processing for active deals and to roll up TEMP_TRAN_DATA rows inserted by the multiple instances of this module and insert/update DAILY_DATA with the summed values and then insert details from TEMP_TRAN_DATA into TRAN_DATA. Income is calculated by retrieving threshold details for each deal component and determining how to perform the calculation (that is, Linear/Scalar, Actuals Earned/Pro-Rate).

Restart/Recovery

A commit will take place after the number of deals records processed is equal to the commit max counter from the RESTART_CONTROL table.

Design Assumptions

N/A

dealday (Daily Posting of Deal Income to Stock & General Ledgers)

Module Name	dealday.pc
Description	Daily Posting of Deal Income to Stock & General Ledgers
Functional Area	Deals
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS208
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch module posts all the deal income records to the Stock Ledger and the Genera Ledger.

This program extracts data inserted by dealinc.pc. In order to simplify this program, a dealday pre function (in prepost.pc) will sum up the data into a temporary table. A dealday post function (in prepost.pc) will copy data to transaction table and then purge temporary tables.

Restart/Recovery

A commit will take place after the number of dept/class/subclass records processed is greater than or equal to the max counter from the RESTART_CONTROL table.

Design Assumptions

N/A

dealfct (Calculates/Update Forecasted Values for Deals)

Module Name	dealfct.pc
Description	Calculates/Update Forecasted Values for Deals
Functional Area	Deals
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS209
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program aggregates income for each item/location and recalculates forecasted values. It maintains forecast periods, deal component totals and deal totals.

After determining which active deals need to have forecast periods updated with actuals, the program will then sum up all the actuals for the deal reporting period and update the table with the summed values and change the period from a forecast period to a fixed period. The program will also adjust either the deal component totals or the remaining forecast periods to ensure that the deal totals remain correct. For each deal, the program will also maintain values held at header level.

Restart/Recovery

A commit will take place after the number of deals records processed is equal to the commit max counter from the RESTART_CONTROL table.

Design Assumptions

N/A

vendinvc (Stage Complex Deal Invoice Information)

Module Name	vendinvc.pc
Description	Stage Complex Deal Invoice Information
Functional Area	Deals
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS122
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The batch module creates records in invoice match staging tables dealing for complex type deals.

The invoicing logic will be driven from the billing period estimated next invoice date for complex deals. The amount to be invoiced will be the sum of the income accruals of the deal since the previous invoice date (or the deal start date for the first collection).

prepost vendinvc pre - truncates STAGE_COMPLEX_DEAL_HEAD and STAGE_COMPLEX_DEAL_DETAIL tables to remove previous days records.

prepost vendinvc post - calls the process_deal_head() function to update est_next_invoice_date of the deal to NULL.

Restart/Recovery

When the max commit point is reached, the data is updated.

I/O Specification

Integration Type	Download from Merchandising
File Name	N /A
Integration Contract	IntCon000009

Records are written to the stage_complex_deal_head and stage_complex_deal_detail tables.

Design Assumptions

N/A

vendinvf (Stage Fixed Deal Invoice Information

Module Name	vendinvc.pc
Description	Stage Complex Deal Invoice Information
Functional Area	Deals
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS123
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The batch module creates records in staging tables dealing for fixed type deals.

The invoicing logic will be driven by the collection dates for fixed deals. The amount to be invoiced will be retrieved directly from fixed deal tables for a given deal date.

prepost vendinvf pre - truncates STAGE_FIXED_DEAL_HEAD and STAGE_FIXED_DEAL_DETAIL tables to remove previous days records.

prepost vendinvf post – calls the process_fixed_deal function to update the status of the fixed deal claim to 'I' (inactive)

Restart/Recovery

Data is committed to the database once the number of transactions processed reaches or exceeds the max_commit_ctr.

I/O Specification

Integration Type	Download from Merchandising
File Name	N /A
Integration Contract	IntCon000009

Records are written to the stage_complex_deal_head and stage_complex_deal_detail tables.

Design Assumptions

N/A

dealcls (Close Expired Deals)

Module Name	dealcls.pc
Description	Close Expired Deals

Functional Area	Deals
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS207
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to close any active deals that have reached their close date. Closed deals are still available in the system for reference and audit purposes, but because the deals are expired, they will not be applied or processed.

Restart/Recovery

N/A

Design Assumptions

N/A

deal_close_job (Close Expired Deals)

Module Name	deal_close_job
Description	Close Expired Deals
Functional Area	Deals
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	N/A

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (DEAL_CLOSE_THREAD) will filter eligible records from deal header (DEAL_HEAD) table that reached their close date. The purpose of this module is to close any active deals that have reached their close date. Closed deals are still available in the system for reference and audit purposes, but as the deals are expired, they will not be applied or processed. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_DEAL_CLOSE_STG.

The Business logic program (DEAL_CLOSE_THREAD) will process all records from the staging table. Using bulk processing, this program will update the records from deal header (DEAL_HEAD) table to "C"losed status. Any existing Deal records from

DEAL_QUEUE table will be re-inserted again through calling FUTURE_COST_EVENT_SQL.ADD_DEALS program. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 6–3 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 6 times a day
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	Threaded by Deal ID

Restart/Recovery

N/A

Key Tables Affected

Table 6–4 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_DEAL_CLOSE_STG	Yes	Yes	No	Yes
DEAL_HEAD	Yes	No	Yes	No
DEAL_QUEUE	Yes	Yes	No	No

dealprg (Purge Closed Deals)

Module Name dealprg.pc

Description Purge Closed Deals

Functional Area Deals

Module Type Admin

Module Technology ProC

Catalog ID RMS212

Wrapper Script rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this batch program is to purge deals after they have been held in the system for the specified number of history months after they are closed. The number of months of history is defined in the PURGE_CONFIG_OPTIONS table in the DEAL_HISTORY_MONTHS column.

The batch program will also delete deal performance tables based on the specified number of history months. This program will not cover PO-specific deals, which will be purged with the PO.

Restart/Recovery

This program has inherent restart/recovery since records that were processed are deleted from the table. As a result, the driving cursor will never fetch the same records again.

Design Assumptions

N/A

deal_purge_job (Purge Closed Deals)

Module Name	deal_purge_job
Description	Purge Closed Deals
Functional Area	Deals
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (DEAL_PURGE_THREAD) will filter eligible records from complex deal header (DEAL_HEAD) and fixed deals (FIXED_DEAL) tables based on its purge criteria from system parameter settings. The Deal History Months (deal_history_months) parameter will determine old/aged deals after they have held in specific number of months after they were closed. PO-specific deals will not be covered in this purge processing. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_DEAL_PURGE_STG.

The Business logic program (DEAL_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from either complex deal related tables or fixed deal related tables depending if indicator is complex deal or not. For Fixed Deals, DELETE_RECORDS_SQL.DEL_FIXED_DEAL is called while complex deals will be processed with call to DELETE_RECORDS_SQL.DEL_DEAL. It

will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 6–5 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	Threaded by Deal ID

Restart/Recovery

N/A

Key Tables Affected

Table 6–6 Key Tables Affected

Table	Select	Insert	Update	Delete
DEAL_HEAD	Yes	No	No	Yes
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_DEAL_PURGE_STG	Yes	Yes	No	Yes
DEAL_HEAD_CFA_EXT	No	No	No	Yes
FIXED_DEAL	Yes	No	No	Yes
DEAL_ITEM_LOC_ EXPLODE	No	No	No	Yes
DEAL ACTUALS_ FORECAST	No	No	No	Yes
DEAL_ITEMLOC_DIV_ GRP	No	No	No	Yes
DEAL_ITEMLOC_DCS	No	No	No	Yes
DEAL_ITEMLOC_ITEM	No	No	No	Yes
DEAL_ITEMLOC_ PARENT_DIFF	No	No	No	Yes
DEAL_COMP_PROM	No	No	No	Yes
DEAL_PROM	No	No	No	Yes

Table 6–6 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
DEAL_THRESHOLD_REV	No	No	No	Yes
DEAL_THRESHOLD	No	No	No	Yes
DEAL_QUEUE	No	No	No	Yes
POP_TERMS_ FULFILLMENT	No	No	No	Yes
POP_TERMS_DEF	No	No	No	Yes
DEAL_DETAIL	No	No	No	Yes
FIXED_DEAL_MERCH_ LOC	No	No	No	Yes
FIXED_DEAL_MERCH	No	No	No	Yes
FIXED_DEAL_DATES	No	No	No	Yes
FIXED_DEAL_GL_REF_ DATA	No	No	No	Yes
KEY_MAP_GL	No	No	No	Yes

deal_actuals_purge_job (Purge Closed Deals Actuals Item/Location)

Module Name deal_actuals_purge_job

Description Purge Closed Deals

Functional Area Deals

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (DEAL_ACTUALS_PURGE_THREAD) will filter eligible records from complex deal header (DEAL_HEAD) and deal actual forecast (DEAL_ ACTUALS_FORECAST) tables based on updated last invoice date that were processed beyond the current date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_DEAL_ACTUALS_ PURGE_STG.

The Business logic program (DEAL_ACTUALS_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from either deal actual item-location (DEAL_ACTUALS_ITEM_LOCATION) table only. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 6–7 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	Threaded by Deal ID

Restart/Recovery

N/A

Key Tables Affected

Table 6–8 Key Tables Affected

Table	Select	Insert	Update	Delete
DEAL_HEAD	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_DEAL_ACTUALS_ PURGE_STG	Yes	Yes	No	No
DEAL_ACTUALS_ITEM_ LOC	No	No	No	Yes
DEAL ACTUALS_ FORECAST	Yes	No	No	No

7

Contracts

Contract batch modules create purchase orders from contracts and purge obsolete contracts. A purchase order created from a contract has two primary differences from all other purchase orders in Merchandising, they are:

- The only impact upon the order is a contract. Bracket costing and deals are not involved in a contract purchase order.
- The cost of an item on the order is predefined in the contract and is held at the item-supplier level.

There are four types of supplier contracts in RMS: A, B, C, and D.

- Type A (Plan/Availability): The contract contains a plan of manufacturing quantity by ready date. Supplier availability is matched to the ready date. Orders are raised against the plan as suggested by replenishment requirements, provided there is sufficient supplier availability. You can also raise manual orders.
- Type B (Plan/No Availability): The contract contains a plan of manufacturing quantity by ready date and dispatch-to location or locations. There are one or more ready dates, which is the date that the items are due at the dispatch-to location. Supplier availability is not required. Orders are raised automatically from the contract based on ready dates.
- Type C (No Plan/No Availability): The contract is an open contract with no production schedule and no supplier availability declared. The contract lists the items that are used to satisfy a total commitment cost. Orders are raised against the contract based on replenishment requirements. The retailer can also raise manual orders.
- Type D (No Plan/Availability): The contract is an open contract with no production schedule. The supplier declares availability as stock is ready. The contract lists the items that are used to satisfy a total commitment cost. Orders are raised against the contract, based on replenishment requirements and supplier availability. The retailer can raise manual orders.

Batch Design Summary

Batch Design Summary

The following batch designs are included in this functional area:

- edidlcon.pc (Download Contracts to Suppliers)
- ediupavl.pc (Upload Item Availability for Type A & D Contracts from Suppliers)
- cntrordb.pc (Create Replenishment Orders for Item/Locations on Type B Contracts)

- cntrprss (Apply Type A, C & D Contracts to Orders Created by Replenishment)
- cntrmain.pc (Contract Maintenance and Purging)
- contract_purge_job (Contract Maintenance and Purging)

edidlcon (Download Contracts to Suppliers)

Module Name	edidlcon.pc
Description	Download Contracts to Suppliers
Functional Area	Contracts
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS45
Wrapper Script	rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Contacts are defined in an Merchandising UI that writes to series of contracts database tables. This program is used to send this contract information to vendors. Only approved contracts that are flagged as EDI contracts are processed by this batch program. The output file of this program contains all records for the supplier contract data which are in approved status.

Restart/Recovery

The logical unit of work for this program is set at the contract number. This program processes one contract number at a time.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000011

Output File Layout

Table 7–1 edidlcon.pc- File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File head descriptor	Char(5)	FHEAD	Describes file line type
	Line Number	Number(10)	0000000001	Sequential file line number
	Gentran ID	Char(4)	'DNCN'	Identifies which translation Gentran uses
	Current date	Char(14)	N/A	Indicates the date that the file was created in YYYYMMD DHH24MISS format

Table 7–1 (Cont.) edidlcon.pc- File Layout

Record Name	Field Name	Field Type	Default Value	Description
THEAD	File head de-scriptor	Char(5)	THEAD	Describes file line type
	Line Number	Number(10)	N/A	Sequential file line number
	Transaction Number	Number(10)	N/A	Sequential transaction number
	Supplier	Number(10)	N/A	Indicates the supplier associated with the contract
	Contract Number	Number(6)	N/A	Indicates the Merchandising contract number
	Contract type	Char(1)	N/A	Type of contract. Valid types are A, B, C or D
	Department	Number(4)	N/A	Indicates the Merchandising department ID for which the contract applies
	Currency code	Char(3)	N/A	Indicates the currency code for the contract
	Total contract cost	Number(20)	N/A	Contains the total cost of the contract; includes 4 implied decimal places
TDETL	File record descriptor	Char(5)	TDETL	Describes file line type
	Line Number	Number(10)	N/A	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number

Table 7–1 (Cont.) ediIcon.pc- File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Item Number Type	Char(6)	N/A	Indicates the type of item number is represented in the file. This corresponds to the item number type defined for items on ITEM_ MASTER
	Item Number	Char(25)	N/A	Contains the unique ID for the item on the contract
	Ref Item Number Type	Char(6)	N/A	Indicates the item number type for the reference number correspondin g to the item number
	Ref Item Number	Char(25)	N/A	Contains the unique ID for the reference number for the item
	Diff1	Char(120)	N/A	Contains the description of Diff1 for the item
	Diff2	Char(120)	N/A	Contains the description of Diff2 for the item
	Diff3	Char(120)	N/A	Contains the description of Diff3 for the item
	Diff4	Char(120)	N/A	Contains the description of Diff4 for the item
	VPN	Char(30)	N/A	Vendor Product Number for the item

Table 7–1 (Cont.) edidlcon.pc- File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Unit cost	Number(20)		Contains the cost of the item on the contract with 4 implied decimal places
	Ready Date	Char(14)		Date on which the items are to be provided by supplier. This field contains only values for contract types of 'A' or 'B'
	Ready Quantity	Number(20)		Quantity contracted with supplier with 4 implied decimal points. This field contains only values for contract types of 'A' or 'B'
	Location Type	Char(2)		Indicates the type of location on the contract - either 'ST' (store) or 'WH' (warehouse). This field contains only values for contract types of 'A' or 'B'
	Location number	Number(10)		Contains a location on the contract. This field contains only values for contract types of 'A' or 'B'

Table 7–1 (Cont.) edidlcon.pc- File Layout

Record Name	Field Name	Field Type	Default Value	Description
TTAIL	File Record descriptor	Char(5)	TTAIL	Describes file line type
	Line Number	Number(10)	N/A	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
FTAIL	File record descriptor	Char(5)	FTAIL	Marks the end of file
	Line number	Number(10)	N/A	Sequential file line number
	Number of lines	Number(10)	N/A	Number of lines in file not counting FHEAD and FTAIL

Design Assumptions

- This module should only be run if contracting is turned on in the system.

ediupavl (Upload Item Availability for Type A & D Contracts from Suppliers)

Module Name	ediupavl.pc
Description	Upload Item Availability for Type A & D Contracts from Suppliers
Functional Area	EDI - Contracts
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS50
Wrapper Script	rmswrap_in_rej.ksh

Schedule

See Oracle Merchandising Batch Schedule.

Design Overview

This module runs to upload supplier availability information, which is a list of the items that a supplier has available. This information is used by Merchandising for type A and D contracts which require supplier availability information. The data uploaded is written to the SUP_AVAIL table.

Restart/Recovery

N/A

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000016

Input File Layout

Table 7–2 ediupavl.pc - File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Describes file line type
	Line number	Number(10)	0000000001	Sequential file line number
	File type	Char(4)	SPAV	N/A
	Create date	Char(14)	N/A	File create date in YYYYMMDDHH24MISS format

Table 7–2 (Cont.) ediupavl.pc - File Layout

Record Name	Field Name	Field Type	Default Value	Description
FDETL	Record descriptor	Char(5)	FDETL	Describes file line type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction number	Number(14)	N/A	Sequential transaction number
	Supplier	Number(10)	N/A	Indicates the supplier for whom the data applies
	Item type	Char(3)	N/A	Indicates the type of item contained in the file. Valid types are 'ITM', 'UPC', or 'VPN'
	Item id	Char(25)	N/A	Unique ID for the item
	Item supplement	Char(5)	N/A	UPC supplement
	Available quantity	Number(12)	N/A	Available quantity including 4 implied decimal places
FTAIL	Record descriptor	Char(5)	FTAIL	Number(10)
	Line number	Number(10)	N/A	Sequential file line number (total # lines in file)
	Number of detail records	Number(10)	N/A	Number of FDETL lines in file

Design Assumptions

- This module will only be run if contracting is turned on in the system.

cntrordb (Create Replenishment Orders for Item/Locations on Type B Contracts)

Module Name cntrordb.pc

Description	Create Replenishment Orders for Item/Locations on Type B Contracts
Functional Area	Contracts
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS232
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module automatically creates replenishment orders for items on an approved, orderable type 'B' contract based on production dates.

Type B (Plan/No Availability) contracts contain a plan of manufacturing quantity by ready date and dispatch-to location or locations. There are one or more ready dates, which is the date that the items are due at the dispatch-to location. Supplier availability is not required. This program automatically writes POs from the contract based on ready dates.

Prepost cntrordb post – updates the system level variable last_cont_order_date to the current vdate

Restart/Recovery

The logical unit of work is contract no. Records are committed to the database when no of records processed reaches commit_max_counter maintained in RESTART_ CONTROL table.

Design Assumptions

- This module should only be run if contracting is turned on in the system.

cntrprss (Apply Type A, C and D Contracts to Orders Created by Replenishment)

Module Name	cntrprss.pc
Description	Apply Type A, C & D Contracts to Orders Created by Replenishment
Functional Area	Contracts
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS202
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module evaluates contracts of type A, C, and D to determine whether an order should be created from the contract. Contracts are ranked so that orders are created off the best contracts first, based on lead-time, cost, contract status (such as,. closed preferred over open), and contract type (such as,. type C are preferred over D). This updates the temporary orders created by the item replenishment extract (rplext) module with the contract and supplier information of the best available contract for each item and populates the repl_results table.

Restart/Recovery

As the item requirements can span across different locations, the logical unit of work varies for each item requirement. For each item requirement, records are committed to the database.

Design Assumptions

- This module should only be run if contracting is turned on in the system.

cntrmain (Contract Maintenance and Purging)

Module Name	cntrmain.pc
Description	Contract Maintenance and Purging
Functional Area	Contracts
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS231
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is used to mark contracts that have reached their end date to completed (for types A and B) or review status (for types C and D). This module also purges contracts that have remained in cancelled, worksheet, submitted, or complete status for a user-defined number of months without any orders and contacts marked for deletion. The number of months is determined by the system parameter for order history months.

Restart/Recovery

This batch program has two processing functions, one for purging and another for updating contracts. The purge function (delete_contracts) deletes and commits records

via arrays whose size is defined in commit max counter while the update function (reset_inactive) updates records in bulk based on the update criteria. The program as a whole is inherently restartable.

Design Assumptions

- This module should only be run if contracting is turned on in the system.

contract_purge_job (Contract Maintenance and Purging)

Module Name	contract_purge_job
Description	Contract Maintenance and Purging
Functional Area	Contracts
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (CONTRACT_MAIN_PURGE_THREAD) will filter eligible records from contract header (contract_header) table based on its purge criteria from system parameter settings. The Order History Months (order_history_months) parameter will determine number of months a contract elapsed in the system comparing current date and contract's status date. Contracts are also considered for deletion when they remained in cancelled, worksheet, submitted, or complete status for a user-defined number of months without any orders and contacts marked for deletion. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_COMP_PRICING_PURGE_STG.

The Business logic program (CONTRACT_MAIN_PURGE) will process all records from the staging table. Using bulk processing, this program will mark contracts that have reached their end date to completed (for types A and B) or review status (for types C and D). It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 7–3 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA

Table 7–3 (Cont.) Scheduling Constraints

Schedule Information	Description
Threading Scheme	Threaded by Contract number

Restart/Recovery

NA

Key Tables Affected

Table 7–4 Key Tables Affected

Table	Select	Insert	Update	Delete
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_CONTRACT_MAIN_ PURGE_STG	Yes	Yes	No	Yes
CONTRACT_HEADER	Yes	No	Yes	Yes
CONTRACT_DETAIL	No	No	No	Yes
CONTRACT_COST	No	No	No	Yes
ORDHEAD	Yes	No	No	No

Design Assumptions

NA

8

Cost Changes

Suppliers often change the cost of items.

Cost is an important factor in individual transactions and many financial calculations in Merchandising. Changes in cost must be reflected in the information stored in Merchandising and pending transactions.

Batch Design Summary

The following batch designs are included in this functional area:

- sccext.pc (Supplier Cost Change Extract)
- ccprg.pc (Cost Change Purge)
- cost_change_purge_job (Cost Change Purge)
- ownership_change_process (Process Scheduled Ownership Change Data)
- ownership_change_purge (Purge Processed and Aged Ownership Change Data)

sccext (Supplier Cost Change Extract)

Module Name	sccext.pc
Description	Apply Pending Cost Changes to Items
Functional Area	Cost Change
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS355
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The sccext module selects supplier cost change records that are set to go into effect the next day and updates the Merchandising item/supplier/country tables with the new cost. The item/location tables are also updated with the new cost if the cost change impacts the primary supplier/country for an item/location, as this is considered a

base cost change. The process also triggers a recalculation of cost and deal application for pending purchase orders.

Restart/Recovery

The logical unit of work for the program is a cost change. The program is also restartable from the last successfully processed cost change record.

Design Assumptions

N/A

ccprg (Cost Change Purge)

Module Name	ccprg.pc
Description	Purge Aged Cost Changes
Functional Area	Cost Change
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS476
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is responsible for removing old cost changes from the system. Cost changes are removed from the system using the following criteria:

- The status of the cost change is Delete, Canceled, or Extracted.
- The status of the price change is Rejected and the effective date of the cost change has met the requirement for the number of days that rejected cost changes are held.

The number of days that rejected cost changes are held is determined by the system parameter Retention of Rejected Cost Changes (RETENTION_OF_REJECTED_COST_CHG).

Restart/Recovery

N/A

Design Assumptions

N/A

cost_change_purge_job (Cost Change Purge)

Module Name	cost_change_purge_job
Description	Purge Aged Cost Changes
Functional Area	Cost Change
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (COST_CHANGE_PURGE_THREAD) will filter eligible records from cost change header (COST_SUSP_SUP_HEAD) table based on its purge criteria from system parameter settings. The Retention of Rejected Cost Changes (retention_of_rejected_cost_chg) parameter will determine the number of days that rejected cost changes are held. It also should met the following criteria:

- The status of the cost change is Delete, Canceled, or Extracted.
- The status of the price change is Rejected and the effective date of the cost change has met the requirement for the number of days that rejected cost changes are held.

These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_COST_CHANGE_PURGE_STG.

The Business logic program (COST_CHANGE_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from cost change header (COST_SUSP_SUP_HEAD) and other related cost change tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 8–1 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by cost change

Restart/Recovery

NA

Key Tables Affected

Table 8–2 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Ys	No	No
B8D_COST_CHANGE_ PURGE_STG	Yes	Yes	No	Yes
COST_SUSP_SUP_HEAD	Yes	No	No	Yes
COST_SUSP_SUP_DETAIL	Yes	No	No	Yes
COST_SUSP_SUP_DETAIL_ LOC	Yes	No	No	Yes

Design Assumptions

NA

ownership_change_process (Process Scheduled Ownership Change Data)

Module Name	ownership_change_process.ksh
Description	Processes scheduled ownership change records from the Ownership Change related tables.
Functional Area	Foundation Data
Module Type	Business Processing
Module Technology	Ksh
Catalog ID	N/A
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program process scheduled and approved ownership change transaction records that are set to go into effect the next day. It updates costing and ownership attributes of item/supplier/country or item/supplier/country/location combinations.

Additionally, primary supplier and country is updated for item/location records. This process will also trigger creation of PO and RTV depending on ownership change type along with relevant tran data postings to account for financial impacts of such

transactions. There will not be any tran data entries if non-inventoried consignment items are involved. In case the item on the ownership change has child items, then the children will also be updated with the same changes as the parent item. On successful completion of the batch, overall status of the Ownership transaction will be updated to 'Processed'.

To account for changes in the item and other transactions that can happen between the time the ownership change was approved and it's picked by the batch for processing; following validations are performed at the time of batch execution. The batch program will fail to process for the ownership change transaction if any of the item/supplier/country/location fails the following validations.

1. The item should not be part of any pack; the exception being when its component of sellable only complex pack or an orderable (non-sellable) buyer complex pack
2. The location should not be included in any other approved Ownership Change transaction for the similar Item/Supplier/Country combination
3. Item should be associated with supplier site and country as specified in the items table.
4. Current Primary supplier site of the location should be same as the one specified in Items table (When the change type is Update Primary Supplier)
5. The item/supplier/country/location combination should not exist on any approved and not closed Purchase Order
6. The item/supplier/country/location combination should not exist on any open Return to Vendor
7. Any replenishment attribute should not be active for item/supplier/country/location combination
8. The item/supplier/country/location combination should not exist on any Cost Change scheduled on or after the effective date of the ownership change
9. Item/supplier/country/location should currently have a purchase type that corresponds to the ownership change type. For example: If Change Type is 'Owned to Consignment' then allow those locations with Purchase Type at Item/Supplier/Country/Location as 'Owned'. Similar treatment for other Change Types.
10. The item/supplier/country location should have a purchase type of either consignment or concession (When the change type is Update Primary Supplier).

    The item/supplier/country/location should have the same purchase type for the current and new supplier/country; either both are consignment or both are concession. (When the change type is Update Primary Supplier).

Restart/Recovery

N/A

Restart/Recovery

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No

Table	Select	Insert	Update	Delete
WH	Yes	No	No	No
STORE	Yes	No	No	No
DEPS	Yes	No	No	No
SUPS	Yes	No	No	No
ADDR	Yes	No	No	No
VAT_REGION	Yes	No	No	No
VAT_CODE_RATES	Yes	No	No	No
VAT_ITEM	Yes	No	No	No
CURRENCIES	Yes	No	No	No
TERMS	Yes	No	No	No
OWNERSHIP_CHANGE_HEAD	Yes	No	Yes	No
OWNERSHIP_CHANGE_DETAIL_ITEM	Yes	No	No	No
OWNERSHIP_CHANGE_DETAIL_LOC	Yes	No	No	No
ITEM_SUPP_COUNTRY_LOC	Yes	No	Yes	No
ITEM_SUPP_COUNTRY	Yes	No	Yes	No
ITEM_LOC	Yes	No	Yes	No
ITEM_LOC_SOH	Yes	No	Yes	No
RTV_HEAD	Yes	Yes	Yes	No
ORDHEAD	Yes	Yes	No	No
INVC_HEAD	Yes	Yes	Yes	No
INVC_DETAIL	Yes	Yes	Yes	No
INVC_MERCH_VAT	Yes	Yes	Yes	Yes
INVC_DETAIL_VAT	No	Yes	No	No
INVC_XREF	No	Yes	No	No
TRAN_DATA	No	Yes	No	No

Design Assumptions

N/A

ownership_change_purge (Purge Processed and Aged Ownership Change Data)

Module Name	ownership_change_purge.ksh
Description	Purges old processed ownership change records from the Ownership Change related tables.
Functional Area	Foundation Data
Module Type	Admin
Module Technology	Ksh

Catalog ID N/A

Wrapper Script rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program purges old and processed records from the ownership change related tables based on the Ownership Change Purge Days system parameter.

The batch program will also archive purged records from each ownership change related tables into respective historical tables.

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
OWNERSHIP_CHANGE_HEAD	Yes	No	No	Yes
OWNERSHIP_CHANGE_DETAIL_ITEM	Yes	No	No	Yes
OWNERSHIP_CHANGE_DETAIL_LOC	Yes	No	No	Yes
OWNERSHIP_CHANGE_DETAIL_ERRORS	Yes	No	No	Yes
OWNCHG_DETAIL_ERRORS_PRG_HIST	No	Yes	No	No
OWNCHG_DETAIL_LOC_PRG_HIST	No	Yes	No	No
OWNCHG_DETAIL_ITEM_PRG_HIST	No	Yes	No	No
OWNERSHIP_CHANGE_HEAD_PRG_HIST	No	Yes	No	No

Design Assumptions

N/A

9

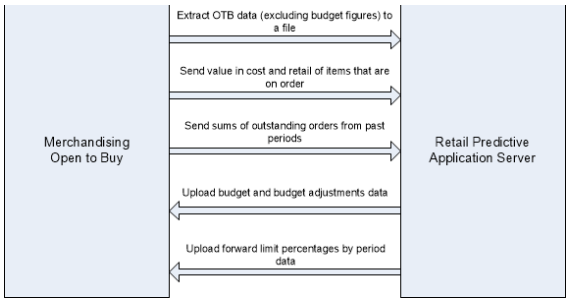
Open To Buy

Open to Buy (OTB) budgets can either be maintained through the Merchandising UI or imported from a planning application.

The programs in this chapter receive OTB data from planning processes and send order information to planning processes and maintain OTB data.

For more information about integration with RPAS and other planning systems, see the section Integration with Oracle Retail Planning.

Figure 9–1 Open To Buy



Batch Design Summary

The following batch designs are included in this functional area:

- otbdnld.pc (Download Current & Future OTB by Subclass)
- otbdlord.pc (Download Summary of Outstanding Orders on OTB by Subclass)
- otbupld.pc (Upload OTB Budget from Planning Systems)
- otbprg.pc (Purge Aged Open To Buy Data)
- otb_purge_job (Purge Aged Open To Buy Data)

otbdnld (Download Current & Future OTB by Subclass)

Module Name	otbdnld.pc
Description	Download Current & Future OTB by Subclass
Functional Area	Open To Buy
Module Type	Integration

Module Technology	ProC
Catalog ID	RMS130
Wrapper Script	rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program will extract current and future Open to Buy data from the OTB table in Merchandising and export it to a flat file for use by an external planning system. All records with an end of week date greater than or equal to today will be sent.

Restart/Recovery

The logical unit of work for the OTBDNLD module is department, class, subclass, and end-of-week date, with a recommended commit counter setting of 10,000. Each time the record counter equals the maximum recommended commit number, an application image array record will be written to the restart_start_array for restart/recovery if a fatal error occurs.

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000031

Output File Layout

Table 9–1 otbdnld.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File Type Record Descriptor	Char (5)	FHEAD	Identifies file record type
	File Line Sequence Number	Number (10)	N/A	Keeps track of the record's position in the file by line number
	File Type Definition	Char (4)	N/A	Identifies file as 'OTB Export'
	File Create Date	Char(14)	N/A	Date the file was created in YYYYMMDD format. Remaining 6 characters are blank
FDETL	File record descriptor	Char(5)	FDETL	Identifies file record type
	File Line Sequence Number	Number (10)		Keeps track of the record's position in the file by line number
	Transaction Set Control Number	Number(14)		Used to force unique file check
	Department	Number(4)		The ID number of a department
	Class	Number(4)		The ID number of a class within the department given
	Subclass	Number(4)		The ID number of a subclass within the class given

Table 9–1 (Cont.) otbdnld.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
	EOW Date	Date		The end of week date for the budgeted period. Format is 'YYYYMMDDHHMMSS'
	Week number	Number(2)		The week number in the month for the budgeted period
	Month number	Number(2)		The month number in the half for the budgeted period
	Half number	Number(5)		The half number for the budgeted period
	Cancel Amount	Number(20)		The total amount cancelled from orders of all order type for the budgeted period; value includes 4 implied decimal places
	N Approved Amount	Number(20)		The amount of approved non-basic (order type N/B) orders for the budgeted period; value includes 4 implied decimal places

Table 9–1 (Cont.) otbdnld.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
	N Receipts Amount	Number(20)		The amount of non-basic (order type N/B) orders due in the budgeted period that have been received; value includes 4 implied decimal places
	B Approved Amount	Number(20)		The amount of approved buyer-replenished basic (order type BRB) orders for the budgeted period; value includes 4 implied decimal places
	B Receipts Amount	Number(20)		The amount of buyer-replenished basic (order type BRB) orders due in the budgeted period that have been received; value includes 4 implied decimal places
	A Approved Amount	Number(20)		The amount of approved auto-replenished basic (order type ARB) orders for the budgeted period; value includes 4 implied decimal places

Table 9–1 (Cont.) otbdnld.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
	A Receipts Amount	Number(20)		The amount of auto-replenih ed basic (order type ARB) orders due in the budgeted period that have been received; value includes 4 implied decimal places
FTAIL	File record descriptor	Char (5)	FTAIL	Identifies file record type
	File Line Sequence Number	Number (10)		Keeps track of the record's position in the file by line number
	Number of lines	Number (10)		Total number of all transaction lines, not including file header and trailer

Design Assumptions

N/A

otbdlord (Download Summary of Outstanding Orders on OTB by Subclass)

Module Name	otbdlord.pc
Description	Download Summary of Outstanding Orders on OTB by Subclass
Functional Area	Open To Buy
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS13
Wrapper Script	rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program runs at the end of the half to delete rows from the OTB table that are at least one half old. The current and previous half's OTB data is retained. The number of days that OTB records are retained by Merchandising is not configurable via a system parameter.

Restart/Recovery

The logical unit of work for the otbdlord module is department/class/subclass. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of the file I/O. The recommended commit counter setting is 10000 records. Each time the record counter equals the maximum recommended commit number, an application image array record will be written to the restart_start_array for restart/recovery if a fatal error occurs.

I/O Specification

Integration Type Download from Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000029

Output File Layout

Table 9–2 otbdlord.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Sequence Number	Number(10)	N/A	Keeps track of the record's position in the file by line number
	File Type Definition	Char(4)	OOEX	Identifies file as 'OTB Out-standing Order Export'
	File Create Date	Char(14)	N/A	Date the file was created in YYYYMMDD format. Remaining six characters are blank.

Table 9–2 (Cont.) otbdlord.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies file record type
	File Line Sequence Number	Number(10)	N/A	Keeps track of the record's position in the file by line number
	Transaction Set Control Number	Number(14)	N/A	Sequence number used to force unique detail record check
	Department	Number(4)	N/A	The number of the department which contains the outstanding order quantity value
	Class	Number(4)	N/A	The number of the class which contains the outstanding order quantity value.

Table 9–2 (Cont.) otbdlord.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
	Subclass	Number(4)	N/A	The number of the subclass which contains the outstanding order quantity value
	N Outstanding Amt	Number(20)	N/A	The amount of outstanding non-basic orders (order type N/B) for past periods; value includes 4 implied decimal places
	B Outstanding Amt	Number(20)	N/A	The amount of outstanding buyer-replenished basic (order type BRB) orders for past periods; value includes 4 implied decimal places
	A Outstanding Amt	Number(20)	N/A	The amount of outstanding auto-replenished basic (order type ARB) orders for past periods; value includes 4 implied decimal places

Table 9–2 (Cont.) otbdlord.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Sequence Number	Number(10)	N/A	Keeps track of the record's position in the file by line number
	Control Number File Line Count	Control Number File Line Count Number(10)	N/A	Total number of all transaction lines, not including file header and trailer

Design Assumptions

N/A

otbupld (Upload OTB Budget from Planning Systems)

Module Name	otbupld.pc
Description	Upload OTB Budget from Planning Systems
Functional Area	Open To Buy
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS132
Wrapper Script	rmswrap_multi_in_rej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this batch module is to accept new and updated open to buy (OTB) budget data from an external planning system. Merchandising supports three types of OTB budgets – those associated with Non-Basic (N/B), Buyer Replenished Basic (BRB) and Auto-Replenished Basic (ARB) orders, as defined by the Order type on Merchandising purchase orders. OTB budgets are created by subclass/end of week date in Merchandising.

Restart/Recovery

Processing of each row is independent and thus if an erroneous record is found during processing; only that record needs to be corrected and reprocessed.

If a record fails validation, it will be written to a rejected record file. This file will facilitate easy reprocessing once the error is fixed by writing the record exactly as it was in the source file.

I/O Specification

Integration Type Upload to Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000033

Input File Layout

Table 9–3 otbupld - Input File

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File head descriptor	Char(5)	FHEAD	Describes file line type
	Line id	Number(10)	0000000001	Sequential file line number
	File Type Definition	Char(4)	'OTBI'	Identifies file as 'OTB Import'
	File Create Date	Char(14)	N/A	The date on which the file was written by external system. The Date is in YYYYMMD DHH24MISS format
	Subclass	Number(4)		The ID number of a subclass within the class given
	Eow Date	Char(14)		The end of week date for the budgeted week in YYYYMMD DHH24MISS format

Table 9–3 (Cont.) otbupld - Input File

Record Name	Field Name	Field Type	Default Value	Description
FDETL	File record descriptor	Char(5)	FDETL	Describes file line type
	Line ID	Number(10)	N/A	Sequential file line number
	Transaction Set Control Number	Number(14)	N/A	Sequence number used to force unique transaction check
	Order Type	Char(1)	N/A	Order type budgeted for: specified as A for ARB, B for BRB, and N for N/B
	Department	Number(4)	N/A	The ID number of a department
	Class	Number(4)	N/A	The ID number of a class within the department given
	Subclass	Number(4)	N/A	The ID number of a subclass within the class given
	Eow Date	Char(14)	N/A	The end of week date for the budgeted week in YYYYMMDDHH24MISS forma
	Budget Amount	Number(20)	N/A	Budgeted amount for the specified order type/week; value includes 4 implied decimal places

Table 9–3 (Cont.) otbupld - Input File

Record Name	Field Name	Field Type	Default Value	Description
FTAIL	File record descriptor	Char(5)	N/A	Marks end of file
	Line ID	Number(10)	Line number in file	Sequential file line number
	Number of lines	Number(10)	Total detail lines	Number of lines in file not counting FHEAD and FTAIL

Design Assumptions

- POs with an Order Type of DSD and Customer Order do not impact open to buy.

otbprg (Purge Aged Open To Buy Data)

Module Name	otbprg.pc
Description	Purge Aged Open To Buy Data
Functional Area	Open To Buy
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS291
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program runs at the end of the half to delete rows from the OTB table that are at least one half old. The current and previous half's OTB data is retained. The number of days that OTB records are retained by Merchandising is not configurable via a system parameter.

Restart/Recovery

There is no restart/recovery in this module. Up to 10,000 records are deleted and committed at a time to avoid excessive rollback space in usage.

Design Assumptions

N/A

otb_purge_job (Purge Aged Open To Buy Data)

Module Name	otb_purge_job
Description	Purge Aged Open To Buy Data
Functional Area	Open To Buy
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (OTB_PURGE_THREAD) will filter eligible records from open-to-buy (OTB) table based on calculated End-of-Week purge date as derived from the current date which is at least one half old. The current and previous half's OTB data is retained and kept in the system. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_OTB_PURGE_STG.

The Business logic program (OTB_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from open-to-buy (OTB) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 9–4 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Thread by rownum

Restart/Recovery

NA

Key Tables Affected

Table 9–5 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No

Table 9–5 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_OTB_PURGE_STG	Yes	Yes	No	Yes
OTB	No	No	No	Yes

Design Assumptions

NA

10

Future Cost

The Future Cost Engine calculates the expected cost of an item/supplier/origin country/location at a given point into the future. These values are used to help in many scenarios (for example, when trying to determine what a margin will be at a point in the future, or when doing investment buying).

The future cost engine can execute as either a synchronous, asynchronous or batch process. The focus of this chapter is the batch processes. To support the discussion of the batch processes, there is general discussion of the engine that is also applicable to the synchronous and asynchronous execution of the engine.

There are also two other programs that are referenced in this chapter related to cost - Pricing Cost Refresh and WAC Refresh. These are both used to refresh materialized views that are used by the WAC Variance report that is displayed by default on the Finance Analyst dashboard.

Future Cost Events

There are three basic events that drive recalculation of FUTURE_COST. They are supplier cost changes, deals, and estimated landed cost components. When these events are added or removed from Merchandising, they impact the calculated values on future cost. These transactions are known as primary events.

There are other events that determine if primary events still apply to a given item/supplier/origin country/location combination. They are reclassifications, merchandise hierarchy changes, organization hierarchy changes, cost zone locations moves, item/cost zones changes, and supplier hierarchy changes. These are secondary events.

There are also two special events that cause new time lines to be created in FUTURE_COST. They are new item loc (when item/locations are ranged) and new item/supplier/country/location relationships (add and remove). These are initialization events.

The ITEM_LOC.PRIMARY_COST_PACK column plays a special roll in costing. When a primary costing pack is defined for an item, that item's costing values are based on the primary_costing_pack not the item its self. When a primary costing pack is added, changed, or removed, this is a primary pack event.

Table 10–1 Cost Events and Cost Event Types

Cost Event	Cost Event Type
Supplier Cost Change	Primary
Deal	Primary

Table 10–1 (Cont.) Cost Events and Cost Event Types

Cost Event	Cost Event Type
ELC Component	Primary
Reclassification	Secondary
Merchandise hierarchy	Secondary
Organization hierarchy	Secondary
Cost zone location moves	Secondary
Item/cost zone changes	Secondary
Supplier hierarchy	Secondary
New Item Location	Initialization
Item/supplier/country/location relationships	Initialization
Primary cost pack	Primary Pack
WF Cost Template	N/A
WF Cost Template Relationship	N/A
Deal Pass through	N/A

Future Cost Engine Run Type Configuration

The Future Cost Engine can be configured by cost event type in one of three ways:

- Synchronous
- Asynchronous
- Batch

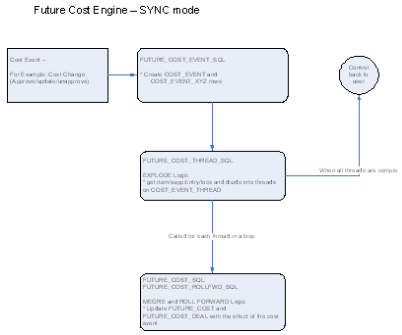
The method to be used by each cost event type is controlled by the configuration defined in the COST_EVENT_RUN_TYPE_CONFIG table.

Synchronous

When running in synchronous mode, the Future Cost Engine is run in the same transaction as the client that calls it. For example if the cost change event is configured to run in synchronous mode, the work done in the Future Cost Engine for the approval of a cost change runs in the same transaction as the flipping of the status of the cost change to 'A' status. That means the user in the form will have a busy cursor until the Future Cost Engine completes.

Cost event types with an EVENT_RUN_TYPE set to 'SYNC' on COST_EVENT_RUN_TYPE_CONFIG will run in synchronous mode.

Figure 10–1 Future Cost Engine - SYNC Mode



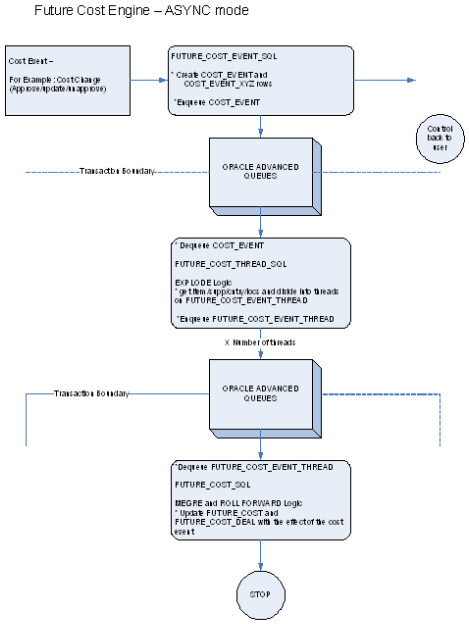
Asynchronous

When running in asynchronous mode, the Future Cost Engine is run in a separate transaction than the client that calls it. For example if the cost change event is configured to run in asynchronous mode, the work done in the Future Cost Engine for the approval of a cost change runs in a different transaction as the flipping of the status of the cost change to 'A' status. This means that control returns to the user in the form while the Future Cost Engine runs in the background.

This is accomplished by using Oracle Advanced Queuing.

Cost event types with an EVENT_RUN_TYPE set to 'ASYNC' on COST_EVENT_RUN_TYPE_CONFIG runs in asynchronous mode.

Figure 10–2 Future Cost Engine - ASYNC Mode



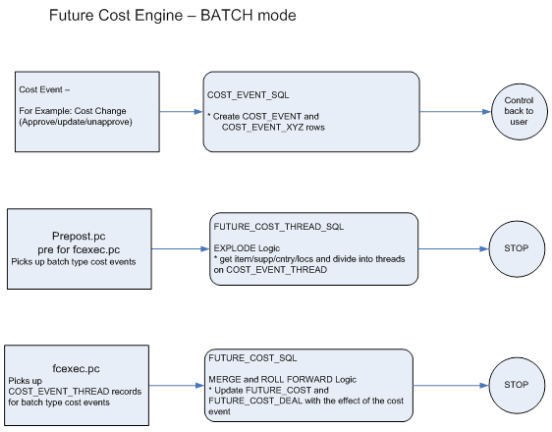
Batch

When running in batch mode, the Future Cost Engine is run during the nightly batch run. For example if the cost change event is configured to run in batch mode, the work done in the Future Cost Engine for the approval of a cost change runs during the next batch run after the approval of the cost change.

Cost event types with an EVENT_RUN_TYPE set to 'BATCH' on COST_EVENT_RUN_TYPE_CONFIG runs in batch mode.

The fcexec.pc batch program and its associated prepost pre job contain logic to run the Future Cost Engine in batch mode.

Figure 10–3 Future Cost Engine - Batch Mode



Future Cost Engine Concurrency Control

Concurrency control is handled in the Future Cost Engine by locking the FUTURE_COST table. The sole job of the Future Cost Engine is maintaining the FUTURE_COST table and its helper DEAL_ITEM_LOC_EXPLODE. The first step in processing is to lock the item/supplier/origin country/location combinations that the cost event covers (after the identification of item/supplier/origin country/location combinations and chunking has been done). If a lock cannot be obtained, another cost event is already processing some of the data that is required. When this occurs the Future Cost Engine stops processing and records the results accordingly and the cost event can be retried at a later time.

Future Cost Engine Error Handling

The COST_EVENT_RESULT table is used to track all runs of the Future Cost Engine whether or not they succeeded. The table records a cost event ID and thread ID, the result code, and any error message that may exist. A special screen is used to search/access the results

Future Cost Engine Threading/Chunking

The Future Cost Engine deals with large amounts of data. Its inputs can vary greatly in size. Its inputs can be one large driver or a group of smaller drivers.

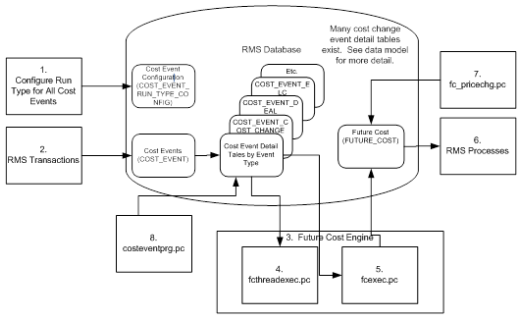
In order to deal with this volume and variation in input a configurable threading/chunking mechanism is built into the Future Cost Engine.

When the transaction control is set to BATCH, the chunks are run in a threaded manner using the Pro*C batch program to coordinate execution.

Future Cost Process

Note: This process focuses on batch runs of the future cost engine.

Figure 10–4 Future Cost Process



- Administrators configure the system (COST_EVENT_RUN_TYPE) to define which cost events types will be processed synchronously, asynchronously or in batch. Configuration by cost event type also determines some threading and chunking parameters.
- Merchandising transactions that should drive future cost recalculation write Cost Events (COST_EVENT and cost event type specific tables).
- Future Cost Engine recalculates future cost

Note: This process flow focuses on batch recalculations, but synchronous or asynchronous processes could easily be substituted in this step.

- fcthreadexec.pc prepares threads for processing
- fcexec.pc recalculates future cost and writes it the future cost table (FUTURE_COST)
- Merchandising processes use future cost information to determine investment buy, margin, and so on.
- fc_pricechg.pc performs special calculation of pricing cost for franchise locations
- costeventprg.pc purges aged cost events from the working cost event tables.

Batch Design Summary

The following batch programs are included in this chapter:

- costeventprg.pc (Purge Aged Cost Events)
- cost_event_purge_job (Purge Aged Cost Events)
- fcexec.pc (Execute Batch Calculation/Recalculation of Future Cost Values)

- fc_pricechg.ksh (Use Pending Price Changes to Drive Recalculation of Pricing Cost for some Franchise Item/Locations)
- fcthreadexec.pc (Prepare Threads for Batch Calculation/Recalculation of Future Cost Values)
- rms_oi_pricecostrefresh.ksh (Refresh MV_PRICING_COST)
- rms_oi_wacvarrefresh.ksh (Update RMS_OI_WAC_VARIANCE_CALC)

costeventprg (Purge Aged Cost Events)

Module Name	costeventprg.pc
Description	Purge Aged Cost Events
Functional Area	Future Cost
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS203
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program purges tables used by the Future Cost calculation engine. Records from the COST_EVENT and its related tables are purged from the system based on the Cost Event History Days (cost_event_hist_days) system parameter.

Restart/Recovery

The logical unit of work is the event type on the COST_EVENT_RUN_TYPE_CONFIG table. Records are deleted serially per event type. Restart recovery is based on deleted records. Restarting on a failed run will resume from records not yet deleted on the prior failed run.

Design Assumptions

N/A

cost_event_purge_job (Purge Aged Cost Events)

Module Name	cost_event_purge_job
Description	Purge Aged Cost Events
Functional Area	Future Cost
Module Type	Admin - Ad hoc
Module Technology	Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (COST_EVENT_PURGE_THREAD) will filter eligible records from cost event (COST_EVENT) table based on its purge criteria from system parameter settings. The Cost Events History Days (cost_event_hist_days) parameter will determine cost events that were old/aged from the creation date. These cost event records should exist from Cost Event Configuration (cost_event_run_type_config) table. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_COST_EVENT_PURGE_STG.

The Business logic program (COST_EVENT_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from specific cost event related tables as per Run Event Type. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 10–2 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Cost Event Process ID

Restart/Recovery

NA

Key Tables Affected

Table 10–3 Key Tables Affected

Table	Select	Insert	Update	Delete
FOUNDATION_UNIT_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_COST_EVENT_PURGE_STG	Yes	Yes	No	Yes

Table 10–3 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
COST_EVENT	No	No	No	Yes
COST_EVENT_RESULT	No	No	No	Yes
COST_EVENT_THREAD	No	No	No	Yes
COST_EVENT_SUPP_ COUNTRY	No	No	No	Yes
COST_EVENT_NIL	No	No	No	Yes
COST_EVENT_PRIM_ PACK	No	No	No	Yes
COST_EVENT_COST_CHG	No	No	No	Yes
COST_EVENT_RECLASS	No	No	No	Yes
COST_EVENT_DEAL	No	No	No	Yes
COST_EVENT_MERCH_ HIER	No	No	No	Yes
COST_EVENT_ORG_HIER	No	No	No	Yes
COST_EVENT_COST_ ZONE	No	No	No	Yes
COST_EVENT_ELC	No	No	No	Yes
COST_EVENT_SUPP_HIER	No	No	No	Yes
COST_EVENT_ITEM_ COST_ZONE	No	No	No	Yes
COST_EVENT_RUN_ TYPE_CONFIG	Yes	No	No	No
COST_EVENT_DEAL_ PASSTHRU	No	No	No	Yes
COST_EVENT_COST RELATIONSHIP	No	No	No	Yes
COST_EVENT_COST_ TMPL	No	No	No	Yes

Design Assumptions

NA

fcexec (Execute Batch Calculation/Recalculation of Future Cost Values)

Module Name	fcexec.pc
Description	Execute Batch Calculation/Recalculation of Future Cost Values
Functional Area	Costing
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS223

Wrapper Script rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The fcexec.pc batch program executes the future cost engine in batch mode. Cost events set up to run in batch mode are threaded in the fcthreadexec.pc batch process and passed to the future cost engine for processing by this program. This program should be always run after the fcthreadexec.pc batch.

This batch program only serves as a wrapper to call the cost engine, the Key Tables Affected section does not list the tables affected by the cost engine. The future cost engine is threaded by item/supplier/country/location.

Restart/Recovery

The logical unit of work for this batch program is the cost_event_process_id on the COST_EVENT table.

Design Assumptions

N/A

fc_pricechg (Use Pending Price Changes to Drive Recalculation of Pricing Cost for some Franchise Item/Locations)

Module Name	fc_pricechg.ksh
Description	Use Pending Price Changes to Drive Recalculation of Pricing Cost for some Franchise Item/Locations
Functional Area	Future Cost
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS497
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script checks for any item/locations that have scheduled price changes for the next day. If there are corresponding item/location rows in the future cost table with the percent-off-retail type template associated then the pricing cost of those future cost records will be recalculated by this program.

Restart/Recovery

N/A

Key Tables Affected

Table 10–4 Key Tables Affected

Table	Select	Insert	Update	Delete
price_hist	Yes	No	No	No
item_master	Yes	No	No	No
wf_cost_relationship	Yes	No	No	No
wf_cost_buildup_tmpl_head	Yes	No	No	No
cost_event_retail_change	Yes	Yes	No	No
Cost_event	Yes	Yes	No	No
Future_cost	Yes	Yes	Yes	No

Design Assumptions

N/A

fcthreadexec (Prepare Threads for Batch Calculation/Recalculation of Future Cost Values)

Module Name	fcthreadexec.pc
Description	Prepare Threads for Batch Calculation/Recalculation of Future Cost Values
Functional Area	Costing
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS230
Wrapper Scripts	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The fcthreadexec.pc batch program is responsible for threading the cost events based on the max_tran_size that is provided in the cost_event_run_type_config table.

This program must always be run before the fcexec batch.

Restart/Recovery

The logical unit of work for this batch program is the cost_event_process_id on the COST_EVENT table.

Design Assumptions

N/A

rms_oi_pricecostrefresh.ksh (Pricing Cost Refresh)

Module Name	rms_oi_pricecostrefresh.ksh
Description	Refreshes the MV_PRICING_COST to reflect the most recent pricing cost of an item/location in FUTURE_COST.
Functional Area	Financial Dashboard
Module Type	Admin
Module Technology	Ksh
Catalog ID	RMS477
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This shell script will refresh the pricing cost materialized view snapshot to reflect the most recent pricing cost for an item/location in from the future cost table. It will in turn insert/update into the WAC variance calculation table to reflect the change in WAC as a result of the change in the average cost of an item/location. This information is used by the WAC Variance report displayed by default in the Finance Analyst dashboard.

Restart/Recovery

N/A

Key Tables Affected

Table 10–5 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_OI_WAC_ VARIANCE_CALC	Yes	No	Yes	No
MV_PRICING_COST	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
CURRENCY_RATES	Yes	No	No	No

Design Assumptions

N/A

rms_oi_wacvarrefresh.ksh (WAC Refresh)

Module Name	rms_oi_wacvarrefresh.ksh
Description	Refreshes the RMS_OI_WAC_VARIANCE_CALC with WAC update on an item/location
Functional Area	Financial Dashboard
Module Type	Admin
Module Technology	Ksh
Catalog ID	RMS478
Runtime Parameters	$UP (database connect string)

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This shell script will refresh the WAC variance calculation table to show the change in WAC for an item/location, based on the change in the item/location's average cost during the day. It is used by the WAC Variance report shown by default in the Finance Analyst dashboard.

Restart/Recovery

N/A

Key Tables Affected

Table 10–6 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_OI_WAC_ VARIANCE_CALC	Yes	No	Yes	No
MV_PRICING_COST	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
CURRENCY_RATES	Yes	No	No	No

Design Assumptions

N/A

11

Invoice Matching

Merchandising stages invoice records to be integrated into the Invoice Matching product. It stages invoice records for Return To Vendor (RTV), Consignment, Deals, Trade Management, Obligations, and Customs Entry.

Batch Design Summary

The following batch designs are included in this functional area:

- edidlinv (Download of Invoice For Invoice Matching)
- invclshp (Close Aged Shipments to Prevent them from Matching Open Invoices)
- invc_ship_close_job (Close Aged Shipments to Prevent them from Matching Open Invoices)
- invprg (Purge Aged Invoices)
- invoice_purge_job (Purge Aged Invoices)

> **Note:** The batch program, saexpim.pc, has a functional connection to this chapter.

edidlinv (Download of Invoice For Invoice Matching)

Module Name	edidlinv.pc
Description	Download of Invoice For Invoice Matching
Functional Area	Invoice Matching
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS127
Wrapper Script	rmswrap_multi_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The EDIDLINV program extracts invoice information from Merchandising invoice tables (INVC_HEAD, INVC_DETAIL) to a flat file. This flat file is used by Invoice Matching to upload invoice data into tables such as IM_DOC_HEAD, IM_INVOICE_DETAIL and IM_DOC_NON_MERCH. This batch program is run daily, extracting invoice records whose invoice date falls on the current vdate.

If the batch is ran adhoc, there may be issues when consignment invoices are generated as the sales process can also run multiple times a day. Invoice information can potentially be not the latest when the extract is generated.

Restart/Recovery

Restart/recovery for this program is set up at the invoice ID and line sequence level. The program resumes writing to file starting on the next line where the previous process ended.

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000024

Output File Layout

Table 11–1 edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Describes file record type. Valid value is FHEAD.k
	Line id	Number(10)	0000000001	Sequential file line number.
	Gentran ID	Char(5)	UPINV	The type of transaction this file represents. Valid value is UPINV
	Current date	Char(14)	N/A	Vdate in YYYYMMD DHH24MISS format.
THEAD	Record descriptor	Char(5)	N/A	Describes file record type. Valid value is THEAD.
	Line id	Number (10)	N/A	Sequential file line number.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Transaction number	Number(10)	N/A	Sequential transaction number. All records within this transaction will also have this transaction number.
	Document Type	Char(6)	N/A	Describes the type of document being uploaded. The document type will determine the types of detail information that are valid for the document upload. Invoice types are held on the codes table under a code type of 'IMIT'.
	Vendor Document Number	Char (50)	N/A	Vendor's document number.
	Group ID	Char(10)	NULL	The Group ID is an informationa l field, which can be used to identify groups of invoices that were transmitted to Invoice Matching together. This is not populated by Merchandisi ng.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Vendor Type	Char(6)	N/A	Type of vendor (either supplier or partner) for this document. Valid values include Bank 'BK', Agent 'AG', Freight Forwarder 'FF', Importer 'IM', Broker 'BR', Factory 'FA', Applicant 'AP', Consolidator 'CO', Consignee 'CN', Supplier Hierarchy Level 1 'S1', Supplier Hierarchy Level 2 'S2', and Supplier Hierarchy Level 3 'S3'. These partner types will be held on the codes table under the code_ type 'PTAL'.
	Vendor ID	Char(10)	N/A	Vendor for this document.
	Vendor Document Date	Char(14)	N/A	Date document was issued by the vendor (in YYYYMMDD24MISSformat).

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Order Number / RTV order number	Number(12)	N/A	Merchandising system order number for this document. Required for merchandise invoices and optional for others. This field can also contain the RTV order number if the RTV flag is 'Y'
	Location	Number(10)	N/A	Merchandising system location for this document.
	Location Type	Char(1)	N/A	Merchandising system location type (either 'S'tore or 'W'arehouse) for this document. Required for merchandise invoices and optional for others.
	Terms	Char(15)	N/A	Terms of this document. If terms are not provided, the vendor's default terms will be associated with this record.
	Due Date	Char(14)	N/A	Terms of this document. If terms are not provided, the vendor's default terms will be associated with this record.
	Payment method	Char(6)	N/A	Method for paying this document.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Currency code	Char(3)	N/A	Currency code for all monetary amounts on this document.
	Exchange rate	Number(20,4)	N/A	Exchange rate *10000 (implied 4 decimal places) for conversion of document currency to the primary currency.
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) total cost amount.
	Total Cost	Number(20,4)	N/A	Total document cost *10000 (implied 4 decimal places), including all items and costs on this document. This value is in the document currency.
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) total vat amount.
	Total VAT Amount	Number(20,4)	N/A	Total VAT amount *10000 (implied 4 decimal places), including all items and costs on this document. This value is in the document currency.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) total quantity amount.
	Total Quantity	Number(12,4)	N/A	Total quantity of items *10000 (implied 4 decimal places) on this document. This value is in EACHES (no other units of measure are supported in Invoice Matching).
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) total discount amount.
	Total Discount	Number(12,4)	N/A	Total discount *10000 (implied 4 decimal places) applied to this document. This value is in the document currency.
	Freight Type	Char(6)	NULL	The freight method for this document. Always blank.
	Paid Ind	Char(1)	N/A	Indicates if this document has been paid.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Multi-Location	Char(1)	N/A	Indicates if this invoice goes to multiple locations.
	Merchandise Type	Char(1)	N/A	Indicates if this invoice is a consignment invoice.
	Deal Id	Number(10)	NULL	Deal Id from Merchandising if this invoice is a deal bill back invoice. Always blank
	Deal Detail Id	Char(10)	NULL	Complex Deal Component Id. Always blank from Merchandising.
	Ref CNR Ext Doc Id	Char(50)	NULL	Reference to the External Id of Credit Note Request associated with this document. Always blank from Merchandising.
	Ref INV Ext Doc Id	Char(50)	NULL	Reference to the External Id of Invoice associated with this document. Always blank from Merchandising.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Deal Approval Indicator	Char(1)	NULL	Indicates if the document on IM_DOC_HEAD is to be created in Approved or Submitted status. Always blank from Merchandising.
	RTV indicator	Char(1)	N/A	Indicates if this invoice is a RTV invoice.
	Custom Document Reference 1	Char(90)	NULL	This optional field is included in the upload file for client customization. No validation will be performed on this field. Always blank from Merchandising.
	Custom Document Reference 2	Char(90)	NULL	This optional field is included in the upload file for client customization. No validation will be performed on this field. Always blank from Merchandising.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Custom Document Reference 3	Char(90)	NULL	This optional field is included in the upload file for client customizatio n. No validation will be performed on this field. Always blank from Merchandisi ng.
	Custom Document Reference 4	Char(90)	NULL	This optional field is included in the upload file for client customizatio n. No validation will be performed on this field. Always blank from Merchandisi ng.
	Cross-referen ce document number	Number(10)	N/A	Document that a credit note is for. Blank for all document types other than merchandise invoices.
TDETL	Record descriptor	Char(5)	N/A	Describes file record type. Valid value is TDETL
	Line id	Number(10)	N/A	Sequential file line number.
	Transaction number	Number(10)	N/A	Transaction number for this item detail record.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	UPC	Char(25)	NULL	UPC for this detail record. Valid item number will be retrieved for the UPC. Always blank from Merchandising.
	UPC Supplement	Number(5)	NULL	Supplement for the UPC. Always blank from Merchandising.
	Item	Char(25)	N/A	Item for this detail record.
	VPN	Char(30)	NULL	Vendor Product Number which can (optionally) be used instead of the Oracle Retail Item Number.
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) Original Document Quantity amount.
	Original Document Quantity	Number(12,4)	N/A	Quantity *10000 (implied 4 decimal places), in EACHES, of the item on this detail record.
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) Original Unit Cost amount.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Original Unit cost	Number(20,4)	N/A	Unit cost *10000 (implied 4 decimal places), in document currency, of the item on this detail record
	Original VAT Code	Char (6)	N/A	VAT code for item.
	Original VAT rate	Number (20,10)	N/A	VAT Rate for the VAT code/item.
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) total allowance. Default is "+" if no allowances exist for this detail record.
	Total Allowance	Number(20,4)	N/A	Sum of allowance details for this item detail record *10000 (implied 4 decimal places). If no allowances exist for this item detail record, value will be 0.
TNMRC	Record descriptor	Char(5)	N/A	Describes file record type.
	Line id	Number (10)	N/A	Sequential file line number.
	Transaction number	Number(10)	N/A	Transaction number for this non-merchandise record.
	Non Merchandise Code	Char(6)	N/A	Non-Merchandise code that describes this cost.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Sign Indicator	Char(1)	N/A	Indicates either a positive (+) or a negative (-) Non Merchandise Amt.
	Non Merchandise Amt	Number(20,4)	N/A	Cost *10000 (implied 4 decimal places) in the document currency.
	Non Merch VAT Code	Char (6)	N/A	VAT Code for Non-Merchandise.
	Non Merch Vat Rate at this VAT code	Number (20, 10)	N/A	VAT Rate corresponding to the VAT code.
	Service Performed Indicator	Char(1)	N/A	Indicates if a service has actually been performed.
	Store	Number(10)	N/A	Store at which the service was performed.
TVATS	File record descriptor	Char(5)		Marks costs at VAT rate line. Valid value is TVATS.
	Line id	Char(10)		Sequential file line number.
	Transaction number	Number(10)		Transaction number for this vat detail record.
	VAT code	Char(6)		VAT code that applies to cost.
	VAT rate	Number (20,10)		VAT Rate corresponding to the VAT code.

Table 11–1 (Cont.) edidlinv.pc - Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Sign Indicator	Char(1)		Indicates either a positive (+) or a negative (-) Original Document Quantity amount.
	Cost at this VAT code	Number (20,4)		Total amount *10000 (implied 4 decimal places) that must be taxed at the above VAT code.
TTAIL	Record descriptor	Char(5)	N/A	Describes file record type. Default value is TTAIL.
	Line id	Number(10)	N/A	Sequential file line number.
	Transaction number	Number(10)	N/A	Transaction number for the transaction that this record is closing.
	Transaction lines	Number(6)	N/A	Total number of detail lines within this transaction.
FTAIL	Record descriptor	Char(5)	N/A	Describes file record type.
	Line id	Number(10)	N/A	Sequential file line number.
	Number of lines	Number(10)	N/A	Total number of lines within this file excluding FHEAD and FTAIL.

Design Assumptions

N/A

invclshp (Close Aged Shipments to Prevent them from Matching Open Invoices)

Module Name	invclshp.pc
Description	Close Aged Shipments to Prevent them from Matching Open Invoices
Functional Area	Invoice Matching
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS252
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program will close all shipments that have remained open for a specified number of days as defined by the 'Close Open Ship Days' system parameter and are not associated with any open invoices. This will be accomplished by setting the invc_match_status on the SHIPMENT table to 'C'losed.

Restart/Recovery

N/A

Design Assumptions

N/A

invc_ship_close_job (Close Aged Shipments to Prevent them from Matching Open Invoices)

Module Name	invc_ship_close_job
Description	Close Aged Shipments to Prevent them from Matching Open Invoices
Functional Area	Invoice Matching
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (INVC_CLOSE_SHIP_THREAD) will filter eligible records from order-shipment (SHIPMENT) and order header (ORDHEAD) tables based on its purge criteria. The Close Open Ship Days (close_open_ship_days) parameter will determine number of days that all shipment records that have remained opened and not associated with any open invoices. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_INVC_CLOSE_SHIP_STG.

The Business logic program (INVC_CLOSE_SHIP) will process all records from the staging table. Using bulk processing, this program will update the records from order-shipment (SHIPMENT) table by setting the invc_match_status to 'C'losed. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 11–2 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 6 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Shipment number

Restart/Recovery

NA

Key Tables Affected

Table 11–3 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_INVC_CLOSE_SHIP_STG	Yes	Yes	No	Yes
ORDHEAD	Yes	No	No	No
SHIPMENT	Yes	No	Yes	No
SHIPSKU	Yes	No	No	No
INVC_HEAD	Yes	No	No	No

Table 11–3 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
INVC_XREF	Yes	No	No	No

invprg (Purge Aged Invoices)

Module Name	Invprg.pc
Description	Purge Aged Invoices
Functional Area	Invoice Matching
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS253
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will purge old posted invoices that have not already been purged by ordprg.pc (which purges invoices associated with an order). This includes all types of invoices-non-merchandise, credit notes, credit note requests, debit memos, and consignment invoices. Regular merchandise invoices will primarily be deleted through the order purge batch (ordprg.pc) but will be deleted by invprg.pc if they still exist in the system. The invoices considered are those older than the number of months defined in the purge_config_options.ORDER_HISTORY_MONTHS column. The age of the invoices will be determined from the match date; if there is no match date, the invoice date will be used.

Note: This program deletes only from the Merchandising invoice tables preceded with 'INVC'.

Restart/Recovery

N/A

Design Assumptions

N/A

invoice_purge_job (Purge Aged Invoices)

Module Name	invoice_purge_job
Description	Purge Aged Invoices
Functional Area	Invoice Matching

Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (INVOICE_PURGE_THREAD) will filter eligible records from invoice header (INVC_HEAD) table based on its purge criteria from system parameter settings. The Order History Months (order_history_months) parameter will determine the number of months older than month ages between current date and invoice match date, invoice date (if match day is not available). These old posted invoices that have not already been purged by Order Purge Job (invoices associated to an order) will be included for deletion. This includes all types of invoices-non-merchandise, credit notes, credit note requests, debit memos, and consignment invoices. Regular merchandise invoices will primarily be deleted through order_purge_job but will be deleted by this job if they still exist in the system. This program deletes only from the RMS invoice tables preceded with 'INVC'. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_INVOICE_PURGE_STG.

The Business logic program (INVOICE_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from invoice-related tables by calling INVC_SQL.DELETE_INVC. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 11–4 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Invoice ID

Restart/Recovery

NA

Key Tables Affected

Table 11–5 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_INVOICE_PURGE_ STG	Yes	Yes	No	Yes
INVC_HEAD	Yes	No	No	Yes
SA_TRAN_HEAD	Yes	No	No	No
SHIPSKU	Yes	No	No	No
INVC_DETAIL	No	No	No	Yes
INVC_NON_MECH	No	No	No	Yes
INVC_MERCH_VAT	No	No	No	Yes
INVC_DETAIL_VAT	No	No	No	Yes
INVC_DISCOUNT	No	No	No	Yes
INVC_TOLERANCE	No	No	No	Yes
ORDLOC_INVC_COST	No	No	Yes	No
INVC_MATCH_QUEUE	No	No	No	Yes

12

Replenishment

Replenishment is a complex business process that monitors stock levels and creates transactions to ensure that stores and WHs have optimal stock levels.

Merchandising supports a number of Replenishment Methods. A Replenishment Method is associated with each item/location being replenished. Each Replenishment Method uses an optimized calculation to determine the correct stock orders to create. Depending on the locations, inventory in the supply chain and other factors, these stock orders can be either Purchase Orders sent to a supplier, Transfers of inventory from WH to store or Allocations.

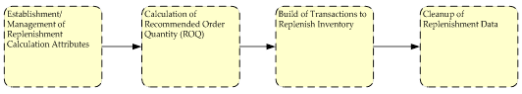
The main purpose of this chapter is to describe the batch processes involved in Replenishment. There is some discussion of user interfaces and database tables involved in the larger Replenishment business process to provide context for the batch processes, but please be aware that the discussion in this chapter of user interfaces and tables not exhaustive.

For additional information about Replenishment, see the Merchandising Functional Library (Doc ID: 1585843.1). Note that the White Papers in this library are intended only for reference and educational purposes and may not reflect the latest version of Oracle Retail software.

Replenishment Sub Processes

Replenishment can be divided into four major sub-processes:

Figure 12–1 Replenishment Sub Processes



1. Establishment/Management of Replenishment Calculation Attributes
    a. Replenishment Calculation Attributes drive how quantities will be calculated. A number of UIs and batch processes maintain this data.
2. Calculation of Recommended Order Quantity (ROQ)
    a. Complex processing determines the Recommended Order Quantity (ROQ) to meet optimal stock level for item/locations based on current stock, forecasts, history, Replenishment Calculation Attributes and other calculation inputs (please note that the inputs and calculations vary depending on the replenishment method selected for each item/location).

   b. If a client uses Investment Buying, additional calculations are performed to determine where additional profitable opportunistic purchases can be made.

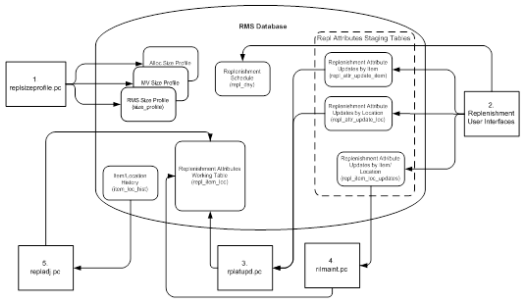
3. Build Transactions to Replenish Inventory
   a. Based on ROQ and Investment Buy, Purchase Orders, Allocations and Transfers are created.
   b. Additional processing optimizes these transactions.
4. Cleanup of Replenishment Data
   a. Cleanup processes purge aged data to ensure good performance.

Establishment/Management of Replenishment Calculation Attributes

Many user and batch processes combine to manage replenishment calculation attributes.

1. replsizeprofile.pc reconciles the size profiles in Merchandising and Allocations and refreshes the size profile materialized view used in replenishment processing.
2. Users create or update assorted replenishment calculation attributes. Data defined by end users includes the schedule the item/location should be reviewed and item/location level attributes. Item/location level attribute changes are written to a series of Replenishment Attribute Staging Tables.
3. rplatupd.pc moves information from the item and location level Replenishment Attribute Staging Tables (repl_attr_update_item and repl_attr_update_loc) to the Replenishment Attributes Working Table (repl_item_loc).
4. rilmaint.pc moves information from the item/loc level Replenishment Attribute Staging Table (repl_item_loc_updates) to the Replenishment Attributes Working Table (repl_item_loc).
5. repladj.pc updates the Replenishment Attributes Working Table (repl_item_loc) for item/locations using the Floating Point Replenishment Method based on history.

Figure 12–2 Managing Replenishment Calculation Attributes



Calculation of Recommended Order Quantity (ROQ)

Many user and batch processes combine to calculate ROQ. Item/Locations follow very different paths through the calculation of ROQ depending on whether they are replenished from inventory (WH to Store via transfer) or from suppliers (via Purchase Order).

1. replroq.ksh determines working net inventory
2. batch_reqext.ksh multithreads reqext.pc

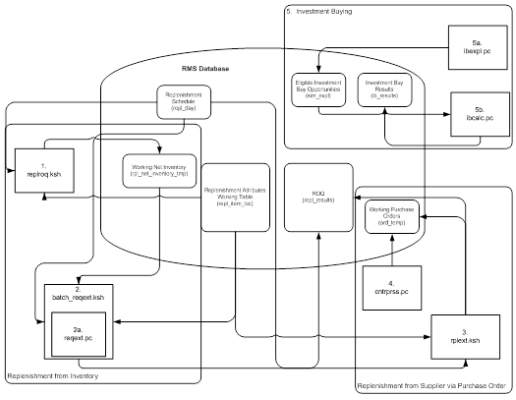
a. reqext.pc uses calculated ROQ in rpl_net_inventory_tmp, franchise order quantity on store_orders, and replenishment attributes to create transfer. Adjusted ROQ is written to repl_results.

Note: Transfers generated by Replenishment will follow the same integration, processing and admin described in the 'Transfers, Allocations and Receiving' described in this volume. Transactions will also be published as described in Volume 2 of the Operations Guide.

---

3. rplext.ksh uses replenishment attributes to determine ROQ for item/locs replenished from suppliers. ROQ is written to repl_results. Working POs are written to ord_temp.
4. If the customer uses Contracts, contracts are evaluated by cntrprss.pc. See the chapter 'Contracts' in this guide for more information.
5. If the customer uses Investment buying
   a. ibexpl.pc determines eligible investment buy opportunities
   b. ibcalc.pc calculates recommended investment buys that will meet the target return-on-investment

Figure 12–3 Calculation of ROQ



Build Transactions to Replenish Inventory

Transactions are built based on ROQ. Additional jobs optimize the resulting POs, Allocations and Transfers.

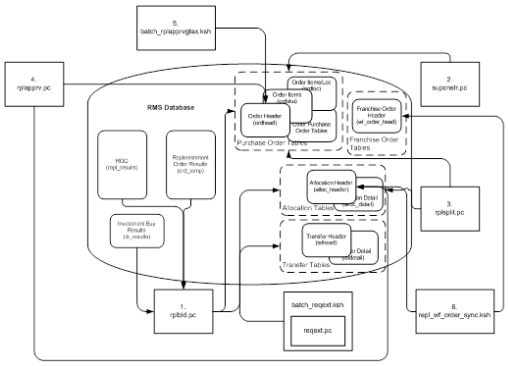
1. rplbld.pc uses ROQ and Investment Buy Results to build Orders
2. supcnstr.pc scales POs based on supplier constraints
3. rplsplit.pc splits POs and Allocations to optimize truck loads
4. rplapprv.pc approves Purchase Orders and Allocations

Note: Once approved, Purchase Orders and Allocations generated by Replenishment will follow the same integration, processing and Admin described in the 'Purchase Orders' and 'Transfers, Allocations and Receiving' described in this volume. Transactions will also be published as described in Volume 2 of the Operations Guide.

---

5. batch_rplapprvgtax.ksh updates tax information (only necessary for GTAX implementations
   a. repl_wf_order_sync.ksh creates appropriate franchise orders for approved allocations created during replenishment

Figure 12–4 Building Transactions to Replenish Inventory



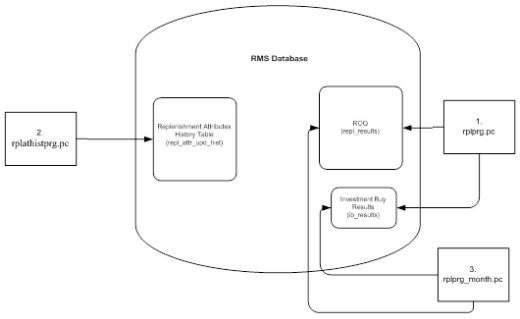
Cleanup Replenishment Data

Replenishment creates large volumes of data. Several programs exist to purge aged replenishment information. Regular purging ensures good batch performance.

> **Note:** Note that all tables discussed in this chapter are not purged by replenishment cleanup jobs. Many replenishment processes clean up their own working tables. The POs, transfers and allocations created by replenishment are purged in their own batch processes.

1. rplprg.pc purges aged ROQ and investment buy results.
2. rplahistprg.pc purges aged replenishment attribute history.
3. rplprg_month.pc purges ROQ and investment buy results.

Figure 12–5 Cleanup Replenishment Data



Batch Design Summary

The following batch designs are included in this chapter:

- replsizeprofile.pc - Update Replenishment Size Profile

- rplatupd.pc - Update Replenishment Calculation Attributes
- rilmaint.pc - Update Replenishment Calculation Attributes by Item/Loc
- repladj.pc - Recalculate Maximum Levels for Floating Point Replenishment
- replroq.ksh - Calculate Net Inventory
- batch_reqext.ksh - Multithreading Wrapper for reqext
- reqext.pc - ROQ Calculation and Distribution for Item/Locs Replenished from WH
- rplext.ksh - ROQ Calculation for Item/Locs Replenished from Supplier
- ibexpl.pc - Determines Eligible Investment Buy Opportunities
- ibcalc.pc - Calculate ROQ for Profitable Investment Buys
- rplbld.pc - Build Replenishment Orders
- supsplit.pc - Split Replenishment Orders Among Suppliers
- rplsplit.pc - Truck Splitting Optimization for Replenishment
- rplapprv.pc - Approve Replenishment Orders
- batch_rplapprvgtax.ksh - Update Replenishment Order Taxes
- repl_wf_order_sync.ksh - Sync Replenishment Franchise Orders
- rplprg.pc - Purge Aged Replenishment Results
- replenishment_purge_job - Purge Aged Replenishment Results
- rplathistprg.pc - Purge Replenishment Attribute History
- rplprg_month.pc - Purge Replenishment Results History by Month
- repl_indctn_purge.ksh - Purge Scheduled Replenishment Induction Staging Tables
- replindbatch.ksh - Upload Replenishment Induction Data
- buyer_wksht_purge_job - Purge Aged Buyer Worksheet Results
- investment_buy_purge_job - Purge Aged Investment Buy Results
- store_orders_purge_job - Purge Aged Store Orders Results

The following batch designs are not included in this chapter, but are related to replenishment as they impact the purchase orders generated by replenishment

- vrplbld.pc – See Purchase Order chapter of this document
- cntrprss.pc – See the Contracts chapter of this document
- supcnstr.pc - See Purchase Order chapter of this document

replsizeprofile (Update Replenishment Size Profile)

Module Name	replsizeprofile.pc
Description	Update Replenishment Size Profile
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC

Catalog ID RMS309

Wrapper Script batch_replsizeprofile.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The batch module will do a total synchronization update of the Merchandising size profile table with data from the Allocation size profile table if the Allocation product is installed. It will also do a complete refresh of the size profile materialized view used by the replenishment attributes update batch and the replenishment attributes screen when size curves are applied to the items being replenished.

Restart/Recovery

N/A

Key Tables Affected

Table 12–1 Key Tables Affected

Table	Select	Insert	Update	Delete
ALC_SIZE_PROFILE	Yes	No	No	No
RMS_SIZE_PROFILE	No	Yes	No	No
MV_SIZE_PROFILE	No	No	Yes	No

Design Assumptions

N/A

rplatupd (Update Replenishment Calculation Attributes)

Module Name rplatupd.pc

Description Update Replenishment Calculation Attributes

Functional Area Replenishment

Module Type Business Processing

Module Technology ProC

Catalog ID RMS313

Wrapper Script rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The batch module reads replenishment attributes from the replenishment attribute item and location tables and processes the item location relationships to determine what replenishment attributes for what locations have to be updated. Replenishment attributes for each item/location are recorded in a separate table. Review cycle information is kept on another table, and the rejected records are written to the mass change rejections table for later reporting.

The pre-processing function of this batch program on prepost truncates records in the mass change rejections table.

The post-processing function of this batch program on prepost locks and deletes records form the various replenishment attributes tables.

Restart/Recovery

The logical unit of work is replenishment attribute id, item, and location. Records will be committed to the database when commit max counter defined in the restart control table is reached.

Key Tables Affected

Table 12–2 Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_ATTR_UPDATE_ ITEM	Yes	No	No	No
REPL_ATTR_UPDATE_ HEAD	Yes	No	No	No
REPL_ATTR_UPDATE_ LOC	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
REPL_ITEM_LOC	Yes	Yes	Yes	Yes
REPL_DAY	No	Yes	No	Yes
ITEM_SEASONS	Yes	Yes	No	No
SYSTEM_OPTIONS	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
PACKITEM	Yes	No	No	No
DEPS	Yes	No	No	No
REPL_ITEM_LOC_ UPDATES	No	Yes	No	Yes
SUB_ITEMS_DETAIL	Yes	No	No	No
MASTER_REPL_ATTR	Yes	Yes	Yes	Yes
REPL_ATTR_UPDATE_ EXCLUDE	Yes	No	No	No
REPL_DAY_UPDATE	Yes	Yes	Yes	Yes
STORE_ORDERS	No	No	No	Yes

Table 12–2 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
PARTNER_ORG_UNIT	Yes	No	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
SUPS	Yes	No	No	No
MV_SIZE_PROFILE	Yes	No	No	No
REPL_ATTR_UPD_HIST	No	Yes	No	No

Design Assumptions

N/A

rilmaint (Update Replenishment Calculation Attributes by Item/Loc)

Module Name	rilmaint.pc
Description	Update Replenishment Calculation Attributes by Item/Location
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS311
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module transfers the replenishment attributes from the item/location replenishment updates table to the item/location replenishment table. Item/location replenishment updates is populated when certain attributes impacting replenishment are modified. These attributes are located across the entire system and are monitored for changes by a series of triggers and modules. Once a change is logged in the item/location replenishment updates table, this program will note the type of change and will update item/location replenishment table appropriately.

Restart/Recovery

The logical unit of work for this batch program is item, change type and location. Records are committed to the database once the maximum commit counter defined in the restart control table is reached.

Key Tables Affected

Table 12–3 Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_ITEM_LOC_ UPDATES	'Yes	No	No	No
REPL_ITEM_LOC	Yes	No	Yes	Yes
REPL_DAY	Yes	No	No	Yes
STORE_ORDERS	No	No	No	Yes
ITEM_MASTER	Yes	No	No	No
PACKITEM	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPP_COUNTRY_ LOC	Yes	No	No	No
MASTER_REPL_ATTR	No	No	No	Yes

Design Assumptions

N/A

repladj (Recalculate Maximum Levels for Floating Point Replenishment)

Module Name	repladj.pc
Description	Recalculate Maximum Levels for Floating Point Replenishment
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS307
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch module recalculates the maximum stock levels for all item-location combinations with replenishment method of 'F' (floating point). The floating model stock method will dynamically calculate an order-up-to-level. The calculated order-up-to-level is used to update the item/location replenishment table.

The maximum model stock (used for calculating order-up-to-level) is derived using the sales history of various periods of time in order to accommodate seasonality as well as trend. The sales history is obtained from the item/location history table.

Restart/Recovery

The module has restart/recovery based on item/ location. Records will be committed to the database when maximum commit counter defined in the restart control table is reached.

Key Tables Affected

Table 12–4 Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_ITEM_LOC	Yes	No	Yes	No
SUB_ITEMS_HEAD	Yes	No	No	No
SUB_ITEMS_DETAIL	Yes	No	No	No
ITEM_LOC_HIST	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
REPL_DAY	Yes	No	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
SUPS	Yes	No	No	No

Design Assumptions

N/A

replroq.ksh (Calculate Net Inventory)

Module Name	replroq.ksh
Description	Calculate Net Inventory
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS308
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module performs the bulk of the logic to process and persist the replenishment data into replenishment net inventory temp table. (The information on this table is extracted by the reqext batch program.)

The wrapper script does the following things:

- Insert records into the staging table and determines the thread id of each record.

- Retrieves the max concurrent thread from to determine the maximum number of concurrent process the wrapper should run at a time.
- Moves the records from a staging table to a temporary table and will calculate the net inventory position and determine the ROQ of items which are on replenishment.

The pre-processing function of this batch program in prepost truncates the records from the replenishment net inventory temp tables, and builds replenishment distribution temp and replenishment allocation in temp tables.

Restart/Recovery

The program processes all items on the replenishment day table for the current day. If the program fails, the program can be restarted and it will process the remaining records on replenishment ROQ table.

Key Tables Affected

Table 12–5 Key Tables Affected

Table	Select	Insert	Update	Delete
DOMAIN_CLASS	Yes	No	No	No
DOMAIN_DEPT	Yes	No	No	No
DOMAIN_SUBCLASS	Yes	No	No	No
REPL_DAY	Yes	No	No	No
REPL_ITEM_LOC	Yes	No	No	No
SVC_REPL_ROQ	Yes	Yes	Yes	Yes
SVC_REPL_ROQ_GTT	Yes	Yes	Yes	Yes
RPL_NET_INVENTORY_TMP	No	Yes	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
STORE_ORDERS	Yes	No	Yes	No
SUPS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

N/A

batch_reqext.ksh (Multithreading Wrapper for reqext)

Module Name batch_reqext.ksh

Description Multithreading Wrapper for reqext

Functional Area Replenishment

Module Type Admin

Module Technology	ksh
Catalog ID	RMS192
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to run the reqext batch program multithreaded.

prepost reqext pre - create the transfer header records for unique combination of Warehouse and Store, stock category, and department.

prepost reqext post – update transfer status to Approved.

Restart/Recovery

N/A - this script only serves as a wrapper for the batch process reqext.pc.

Key Tables Affected

Table 12–6 Key Tables Affected

Table	Select	Insert	Update	Delete
ALL_TAB_PARTITIONS	Yes	No	No	No
RESTART_CONTROL	Yes	No	No	No

Design Assumptions

N/A

reqext (ROQ Calculation and Distribution for Item/Locs Replenished from WH)

Module Name	reqext.pc
Description	ROQ Calculation and Distribution for Item/Locations Replenished from Warehouse
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS310
Runtime Parameters	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module performs the automatic replenishment of items from warehouses to stores. It runs through every item-store combination set to be reviewed on the current day, and calculates the quantity of the item, known as the recommended order quantity (ROQ) that needs to be transferred to the store (if any). In addition, it distributes this ROQ over any applicable alternate items associated with the item.

Once the transfer quantity of an item has been calculated, transfers are created and records are written to the replenishment results tablebased on the replenishment order control indicator. For franchise stores, separate transfers are created based on the need date and will be linked back to a Franchise Order through the Franchise Order Number field.

This batch will also insert records into the respective tables for supporting the localization feature. This will be applicable only if localizations are enabled.

The pre-processing function of this batch in prepost will create transfer header records for unique combinations of warehouse and store, stock category and department.

The post-processing function of this batch will update the transfer status to Approved.

Restart/Recovery

The logical unit of work is an item/source warehouse. Restart/recovery is achieved implicitly because item/location replenishment records that have been processed are updated with a last review date and only records that have not been reviewed today will be picked up by the driving cursor again. Records will be committed to the database when the maximum commit counter defined in the restart control table is reached. During the night run the batch processed only those store order records with delivery slot. The review dates are not updated during day run. During night all the records are processed irrespective of the delivery slots.

Key Tables Affected

Table 12–7 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_LOC	Yes	No	No	No
ITEM_LOC_SOH	No	No	Yes	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
PACKHEAD	Yes	No	No	No
PACKITEM	Yes	No	No	No
PACKSTORE_HIST	Yes	No	No	No
PERIOD	Yes	No	No	No
REPL_DAY	Yes	No	No	No
REPL_ITEM_LOC	Yes	No	Yes	No
REPL_RESULTS	No	Yes	No	No

Table 12–7 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
RPL_NET_INVENTORY_TMP	Yes	No	No	No
STORE	Yes	No	No	No
SUB_ITEMS_DETAIL	Yes	No	No	No
SUB_ITEMS_HEAD	Yes	No	No	No
SUPS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
TSFDETAIL	Yes	Yes	Yes	No
TSFHEAD	Yes	Yes	No	No
WH	Yes	No	No	No
STORE_ORDERS	Yes	No	Yes	No
WF_ORDER_HEAD	No	Yes	No	No
WF_ORDER_DETAIL	Yes	Yes	No	No
DELIVERY_SLOT	Yes	No	No	No
ADDR	Yes	No	No	No
COMPHEAD	Yes	No	No	No
OUTLOC	Yes	No	No	No
L10N_DOC_DETAILS_GTT	Yes	Yes	No	No
MV_L10N_ENTITY	Yes	No	No	No
COUNTRY_ATTRIB	Yes	No	No	No
L10N_PKG_CONFIG	Yes	No	No	No
ORDHEAD_L10N_EXT	No	Yes	No	No
TSFHEAD_L10N_EXT	No	Yes	No	No
MRT_L10N_EXT	No	Yes	No	No

Design Assumptions

N/A

rplext.ksh (ROQ Calculation and Distribution for Item/Locs Replenished from Supplier

Module Name	rplext.ksh
Description	ROQ Calculation and Distribution for Item/Locs Replenished from Supplier
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	KSH

Catalog ID RMS315

Wrapper Script rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Vendor Replenishment Extraction, which uses bulk processing logic, is the driving program for the replenishment process. It cycles through every item-location combination that is ready to be reviewed on the current day, and calculates the quantity of the item that needs to be ordered to the location. The program then writes these temporary order line items to the temporary order table and replenishment results. The temporary order table is later reviewed by the batch in its evaluation of orders against contract types A, C, D, whereas replenishment results is processed by build replenishment orders.

The wrapper script does the following things:

- Calls the function will insert records into the replenishment ROQ table and determines the thread id of each record.
- Retrieves the max concurrent thread from configuration table to determine the maximum number of concurrent processes the wrapper should run at a time.
- For each thread, call the function that will move the records from the replenishment ROQ table to the replenishment ROQ global temporary table and the processed records will be inserted to the temporary order and replenishment results tables.

The pre-processing function for this program in the prepost batch truncates records form the temporary order table and the missed order table.

The post-processing function for this program truncates the replenishment distro temp and replenishment allocation in temp table.

Restart/Recovery

If the program fails, the program can be restarted and it will process the remaining records on replenishment table.

Locking Strategy

STORE_ORDER table records are locked while calculating ROQ.

Performance Considerations

The values on RMS_PLSQL_BATCH_CONFIG can be change to alter the behavior of the chunking and threading process.

MAX_CHUNK_SIZE - determines the maximum number of rows that should be processed for a given thread. Currently, this is set to 10000.

MAX_CONCURRENT_THREAD - determines the maximum number of parallel threads for a given run. Currently, this is set to 32.

Key Tables Affected

Table 12–8 Key Tables Affected

Table	Select	Insert	Update	Delete
DOMAIN_CLASS	Yes	No	No	No
DOMAIN_DEPT	Yes	No	No	No
DOMAIN_SUBCLASS	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
PERIOD	Yes	No	No	No
REPL_DAY	Yes	No	No	No
REPL_ITEM_LOC	Yes	No	Yes	No
STORE	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
WH	Yes	No	No	No
SUPS	Yes	No	No	No
SUP_INV_MGMT	Yes	No	No	No
ORD_TEMP	No	Yes	No	No
REPL_RESULTS	No	Yes	No	No

Design Assumptions

N/A

ibexpl (Determines Eligible Investment Buy Opportunities)

Module Name	ibexpl.pc
Description	Determines Eligible Investment Buy Opportunities
Functional Area	Investment Buy
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS250
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The batch program pre-qualifies investment buy (IB) eligible wh/dept and IB eligible supp/dept/locs.

The warehouse/department table holds IB parameters at the warehouse or at the warehouse/department level. If there are IB parameters defined at the warehouse/department level, they are used. If there are no IB parameters defined at

the warehouse/department level, the IB parameters at the warehouse level are used. If IB parameters are not defined at either level, then system level IB parameters are used. The first part of this program sends IB parameters to the warehouse/department level no matter what level they are held at in the database. The results are written to the warehouse/department explode table.

Next the warehouse/department explode table is combined with supplier inventory management data to get the final list of all eligible supplier/department/locations. The supplier inventory management data determines whether or not a given sup/dept/loc combo is IB eligible. The main problem is that this table can store information at different levels depending upon the supplier's inventory management level. Valid options for this level are:

The main problem is that this table can store information at different levels depending upon the supplier's inventory management level.

Valid options for this level are:

- Supplier (S)
- Supplier/department (D)
- Supplier/location (L)
- Supplier/department/location (A)

If the record is not found at the defined level, it needs to look up the hierarchy as shown below, up to the highest level (supplier). If no record exists as the supplier level, it is not IB eligible.

- Supplier
- Supplier/department -> Supplier
- Supplier/location -> Supplier
- Supplier/department/location -> Supplier/department ' Supplier

The second part of this program explodes the supplier inventory management data down to the supplier/department/location level by filling in the implied rows. The exploded supplier inventory management information is only done for IB eligible warehouse/department combinations from the warehouse/department explode table. The results are placed on the SIM explode table.

Restart/Recovery

N/A

Key Tables Affected

Table 12–9 Key Tables Affected

Table	Select	Insert	Update	Delete
WH	Yes	No	No	No
DEPS	Yes	No	No	No
WH_DEPT	Yes	No	No	No
SUP_INV_MGMT	Yes	No	No	No
SUPS	Yes	No	No	No
WH_DEPT_EXPL	Yes	Yes	No	Yes

Table 12–9 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
TERMS	Yes	No	No	No
SIM_EXPL	No	Yes	No	Yes
SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

N/A

buyer_wksht_purge_job (Purge Aged Buyer Worksheet Results)

Module Name buyer_wksht_purge_job

Description Purge Aged Buyer Worksheet Results

Functional Area Replenishment

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (BUYER_WKSHT_PURGE_THREAD) will filter eligible records from buyer worksheet manual results (BUYER_WKSHT_MANUAL) table based on its purge criteria from system parameter settings. The Replenishment Result Purging Cycle (repl_results_purge_days) parameter will determine those unneeded records that are older than predetermined number of days from its creation date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_BUYER_WKSHT_PURGE_STG.

The Business logic program (BUYER_WKSHT_PURGE) will process all records from the staging table. Using bulk processing, this program will delete old records from buyer worksheet manual (BUYER_WKSHT_MANUAL) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 12–10 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 6 hours interval - 2 times a day
Scheduling Considerations	NA
Pre-Processing	NA

Table 12–10 (Cont.) Scheduling Constraints

Schedule Information	Description
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 12–11 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_BUYER_WKSHT_PURGE_STG	Yes	Yes	No	Yes
BUYER_WKSHT_MANUAL	No	No	No	Yes

Design Assumptions

NA

ibcalc (Calculate ROQ for Profitable Investment Buys)

Module Name	ibcalc.pc
Description	Calculate ROQ for Profitable Investment Buys
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS249
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The batch program is the calculation engine for investment buy processing. It identifies investment buy (IB) opportunities and calculates recommended order quantities (ROQs) that will meet the target return-on-investment (ROI)

This module will calculate forward buy opportunities using:

- Carrying costs
- Ordering parameters
- Deals – future and expiring
- Cost changes – future
- Forecasts
- Inventory levels
- Target ROI (return on investment)

The deals and cost change components will be contained on the future cost table. This table will hold a record for each future date that has a costing event (for example, a cost change, deal activation/deactivation). This process utilizes the default costing bracket and default deal thresholds in the calculations.

The pre-processing for this batch in the prepost program sets the status of investment buy from 'W' (worksheet) to 'U' (unprocessed).

Restart/Recovery

The logical unit of work is item and location combination.

Key Tables Affected

Table 12–12 Key Tables Affected

Table	Select	Insert	Update	Delete
FUTURE_COST	Yes	No	No	No
SIM_EXPL	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
REPL_ITEM_LOC	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
PACKITEM	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPP_COUNTRY_LOC	Yes	No	No	No
ITEM_SUPP_COUNTRY_DIM	Yes	No	No	No
SUPS	Yes	No	No	No
SUB_ITEMS_DETAIL	Yes	No	No	No
SUB_ITEMS_HEAD	Yes	No	No	No
UOM_CONVERSION	Yes	No	No	No
WH	Yes	No	No	No
IB_RESULTS	No	Yes	No	No

Design Assumptions

N/A

rplbld (Build Replenishment Orders)

Module Name	rplbld.pc
Description	Build Replenishment Orders
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS314
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program builds Merchandising orders from recommended order quantities (ROQ) generated by the replenishment extract and investment buy calculation processes. The apply type A, C & D contracts to orders created by replenishment batch associates contracts with the ROQs created by the ROQ calculation and distribution for item/locations replenished from supplier program. These ROQs are placed on a temporary table by the replenishment extract and investment buy calculation processes. All records on the temporary tables are processed by this batch each night. These temporary table records are placed into logical groups, and a Merchandising order is created for each logical group.

In order to be placed in the same order group, the item/location ROQs from the temporary tables must share a common supplier, have the same order_status ('W'orksheet or 'A'pproved), and be on the same contract (or not be associated with a contract). Depending on flags on the order inventory management table, two other criteria can be used for splitting order groups. First, if the inventory management level is set to 'D'ept, only items in a single department are allowed in an ordering group. Secondly, the single location indicator can be set to 'Y'es. If this is the case, only one location is allowed per ordering group. Finally, an item may only exist in an ordering group with a single origin country. When an item/location ROQ temporary table record is encountered with a different origin country than the one it exists with in the current ordering group, it is placed in a different ordering group.

To assist the recalculation and order scaling processes of replenishment ROQs, the replenishment results record, associated with the order temporary table record being processed, is updated with the order number and allocation number that the order temporary table record was placed with. Investment Buy results is also updated with the order number.

If the location to be replenished is a Franchise location and the replenishment Order Control is Semi-Automatic or Automatic, Franchise POs will be created per Costing Location/Location. Associated Franchise Orders will also be created.

Restart/Recovery

The logical unit of work is supplier, contract number, and order status. Records will be committed to the database when commit max counter defined in the restart control table is reached.

Key Tables Affected

Table 12–13 Key Tables Affected

Table	Select	Insert	Update	Delete
ORD_TEMP	Yes	No	No	No
REPL_RESULTS	Yes	No	Yes	No
WH	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
IB_RESULTS	Yes	No	Yes	No
CONTRACT_HEADER	Yes	No	Yes	No
CONTRACT_DETAIL	Yes	No	Yes	No
ORDSKU	Yes	Yes	No	No
ORDLOC	Yes	Yes	No	No
ALLOC_HEADER	No	Yes	No	No
ALLOC_DETAIL	No	Yes	No	No
ITEM_LOC	Yes	No	No	No
ORDHEAD	Yes	Yes	Yes	No
ORD_INV_MGMT	Yes	Yes	Yes	No
ORDLC	No	Yes	No	No
ITEM_SUPP_COUNTRY_LOC	No	No	No	No
ITEM_SUPP_COUNTRY	No	No	Yes	No
BUYER_WKSHT_MANUAL	No	No	Yes	No
L10N_DOC_DETAILS_GTT	Yes	Yes	No	No
MV_L10N_ENTITY	Yes	No	No	No
COUNTRY_ATTRIB	Yes	No	No	No
L10N_PKG_CONFIG	Yes	No	No	No
TSFHEAD	Yes	No	No	No
ORDHEAD_L10N_EXT	No	Yes	No	No
TSFHEAD_L10N_EXT	No	Yes	No	No
MRT_L10N_EXT	No	Yes	No	No
FM_SYSTEM_OPTIONS	Yes	No	No	No
WF_ORDER_HEAD	No	Yes	No	No
WF_ORDER_DETAIL	No	Yes	No	No

Design Assumptions

N/A

supsplit (Split Replenishment Orders Among Suppliers)

Module Name	supsplit.pc
Description	Split Replenishment Orders Among Suppliers
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS370
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program splits replenishment orders among different suppliers based on the supplier distribution ratio setup for an item/location on replenishment. It only applies to Direct to Store and Crossdock replenishments where a purchase order will be created from a supplier.

Restart/Recovery

The logical unit of work for this program is set at item level. Records will be committed to the database when commit max counter defined in the restart control table is reached.

Key Tables Affected

Table 12–14 Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_ITEM_LOC_SUPP_DIST	Yes	No	No	No
ORD_TEMP	Yes	Yes	No	Yes
REPL_RESULTS	Yes	Yes	No	Yes
ITEM_MASTER	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPP_COUNTRY_LOC	Yes	No	No	No

Design Assumptions

N/A

rplsplit (Truck Splitting Optimization for Replenishment)

Module Name	rplsplit.pc

Description	Truck Splitting Optimization for Replenishment
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS318
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this program is to select all the orders eligible for truck splitting, which are created by the replenishment programs. The orders that are eligible will be sent into the truck splitting logic and the resulting orders will be created.

The orders, which will be eligible for splitting, are as follows:

- The order must have been created today by replenishment with the order approve indicator set to Yes.
- The order must not have been already split.
- The order must be a single location order and the location must be a warehouse.
- The order must not have any allocations associated.

Orders will only be split if they meet criteria for splitting as defined in the supplier inventory management parameters.

Restart/Recovery

The logical unit of work for this program is set at order level. Records will be committed to the database when commit max counter defined in the restart control table is reached.

Key Tables Affected

Table 12–15 Key Tables Affected

Table	Select	Insert	Update	Delete
ORDHEAD	Yes	Yes	Yes	No
ORDSKU	Yes	Yes	No	Yes
ORDLOC	Yes	Yes	No	Yes
ORD_INV_MGMT	Yes	Yes	Yes	Yes
ITEM_MASTER	Yes	No	No	No
WH	Yes	No	No	No
V_RESTART_SUPPLIER	Yes	No	No	No
ALLOC_HEADER	Yes	Yes	No	Yes
ALLOC_DETAIL	Yes	Yes	No	Yes

Table 12–15 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ALLOC_CHRG	No	No	No	Yes
ORDHEAD_REV	No	No	No	Yes
ORDSKU_REV	No	No	No	Yes
ORDLOC_REV	No	No	No	Yes
ORDLOC_WKSHT	No	No	No	Yes
ORDLOC_DISCOUNT	No	No	No	Yes
ORDCUST	No	No	No	Yes
ORDLC	No	No	No	Yes
DEAL_COMP_PROM	No	No	No	Yes
DEAL_ITEMLOC	No	No	No	Yes
DEAL_THRESHOLD	No	No	No	Yes
DEAL_DETAIL	No	No	No	Yes
DEAL_QUEUE	No	No	No	Yes
DEAL_CALC_QUEUE	No	No	No	Yes
DEAL_HEAD	No	No	No	Yes
REPL_RESULTS	No	No	No	Yes
REV_ORDERS	No	No	No	Yes
ITEM_LOC	Yes	No	No	No
ITEM_SUPP_COUNTRY_ LOC	Yes	No	No	No
CONTRACT_DETAIL	No	No	Yes	No
CONTRACT_HEAD	No	No	Yes	No
BUYER_WKSHT_ MANUAL	No	No	Yes	No
IB_RESULTS	No	No	Yes	No
L10N_DOC_DETAILS_GTT	Yes	No	No	Yes
MV_L10N_ENTITY	Yes	No	No	No
COUNTRY_ATTRIB	Yes	No	No	No
L10N_PKG_CONFIG	Yes	No	No	No
TSFHEAD	Yes	No	No	No
ORDHEAD_L10N_EXT	No	Yes	No	No
TSFHEAD_L10N_EXT	No	Yes	No	No
MRT_L10N_EXT	No	Yes	No	No
FM_SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

N/A

rplapprv (Approve Replenishment Orders)

Module Name	rplapprv.pc
Description	Approve Replenishment Orders
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS300
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program looks at all replenishment, vendor and contract orders created during the nightly batch run to determine if they can be approved. These orders are compared with any vendor minimums that may exist. Orders that do not meet the vendor minimums are either deleted or placed in worksheet status. A flag, held at the supplier inventory management level, determines what action is taken on orders that fail minimums. Vendor generated orders are not subject to these minimum checks.

Vendor minimums can be held at the order, item, or location level. Order and location level minimums are held on the supplier inventory management table. There is a flag that determines if they are applied at the order level or at the location level. Vendor minimums at the item level are held on the item/supplier/country table.

When an order fails the minimums, and the flag is set to 'N', a failure at any level causes the order to be placed in worksheet status. When the flag is 'Y', a failure at the location level causes the offending location to be deleted; a failure at the item level causes the problematic item to be deleted; and a failure at the order level caused the entire order to be deleted.

For any orders that fail vendor minimums when the flag is set to 'Y', a record is written to the supplier minimum failures table for reporting purposes. This table is purged during the pre-processing of this batch program.

After order records are updated, any applicable deals, brackets and allowances are applied to the orders by subsequent processes. Open to buy is then updated for any orders built in approved status. If any orders are contract orders, the contract amounts are updated as well to reflect any order record deletions.

If any orders are Franchise POs, the associated Franchise Orders are also approved if they pass the credit check. If they fail the credit check, both Franchise POs and orders will remain in Worksheet.

An order may not pass vendor minimum checks assuming that the vendor minimum checks are performed for a physical warehouse. If the vendor minimum is not met for a physical location, all the virtual warehouses on the order within the physical warehouse will need to be removed along with associated allocations.

The pre-processing function for this batch program on prepost truncates the supplier minimum failures table.

Restart/Recovery

The logical unit of work is order number. Records will be committed to the database when commit max counter defined in the restart control table is reached.

Key Tables Affected

Table 12–16 Key Tables Affected

Table	Select	Insert	Update	Delete
ORDHEAD_LOCK	No	No	No	Yes
ORDHEAD	Yes	No	Yes	Yes
ORDLOC	Yes	No	No	Yes
ORDSKU	Yes	No	No	Yes
ORD_INV_MGMT	Yes	No	Yes	Yes
DEAL_CALC_QUEUE	No	Yes	Yes	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
SUPS_MIN_FAIL	No	Yes	No	Yes
ALLOC_HEADER	Yes	No	Yes	Yes
ALLOC_DETAIL	No	No	No	Yes
CONTRACT_HEADER	Yes	No	Yes	No
OTB	No	No	Yes	No
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
WH	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
SUPS	Yes	No	No	No
REPL_APPRV_GTAX_QUEUE	No	Yes	No	No
ORDHEAD_CFA_EXT	No	No	No	Yes
WF_ORDER_HEAD	Yes	No	Yes	No

Design Assumptions

N/A

batch_rplapprvgtax.ksh (Update Replenishment Order Taxes)

Module Name	batch_rplapprvgtax.ksh
Description	Update Replenishment Order Taxes
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ksh

Catalog ID RMS194

Wrapper Script N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script calls a function to enable parallel execution via multiple thread calls compute taxes for approved replenishment orders. Computed taxes are inserted/updated into the order tax breakup table.

This batch should be run only for Global Tax (GTAX) configuration.

Restart/Recovery

The logical unit of work is a set of purchase orders. Purchase order numbers in the replenishment approval GTAX queue table are assigned a thread number given the number of slots.

The same table drives the restart and recovery as well. Purchase orders in a thread that successfully complete execution are deleted from replenishment approval GTAX queue. Any restart after a fatal error will include the failed purchase order numbers when assigning new threads.

Key Tables Affected

Table 12–17 Key Tables Affected

Table	Select	Insert	Update	Delete
ORD_TAX_BREAKUP	Yes	Yes	Yes	No
REPL_APPRV_GTAX_QUEUE	Yes	No	No	Yes
ORDLOC	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
MV_CURRENCY_CONVERSION_RATES	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
V_PACKSKU_QTY	Yes	No	No	No
ADDR	Yes	No	No	No
STATE	Yes	No	No	No
COUNTRY	Yes	No	No	No
COUNTRY_TAX_JURISDICTION	Yes	No	No	No
V_BR_COUNTRY_ATTRIB	Yes	No	No	No
V_BR_SUPS	Yes	No	No	No
V_BR_STORE_FISCAL_CLASS	Yes	No	No	No

Table 12–17 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
V_BR_STORE_REG_NUM	Yes	No	No	No
V_BR_WH_REG_NUM	Yes	No	No	No
V_BR_ITEM_FISCAL_ ATTRIB	Yes	No	No	No
ORDLOC_EXP	Yes	No	No	No
ELC_COMP	Yes	No	No	No
ORDLOC_DISCOUNT	Yes	No	No	No
VAT_CODES	Yes	No	No	No
FM_FISCAL_UTILIZATION	Yes	No	No	No
V_BR_ORD_UTIL_CODE	Yes	No	No	No

Design Assumptions

This program should only be run in Global Tax (GTAX) installations.

repl_wf_order_sync.ksh (Sync Replenishment Franchise Orders

Module Name	repl_wh_order_sync.ksh
Description	Sync Replenishment Franchise Orders
Functional Area	Replenishment
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS306
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module will serve as the wrapper for a package function which will check the crossdock orders created during replenishment and create franchise order records for any allocations where the destination location is a franchise store.

Restart/Recovery

N/A

Key Tables Affected

Table 12–18 Key Tables Affected

Table	Select	Insert	Update	Delete
ALLOC_HEADER	Yes	No	Yes	No
ALLOC_DETAIL	Yes	No	Yes	No
STORE	Yes	No	No	No
WF_CUSTOMER	Yes	No	No	No
WF_ORDER_HEAD	Yes	Yes	Yes	No
WF_ORDER_DETAIL	Yes	Yes	Yes	Yes
ITEM_MASTER	Yes	No	No	No
STORE	Yes	No	No	No
WF_ORDER_AUDIT	Yes	No	No	Yes
FUTURE_COST	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPP_COUNTRY_DIM	Yes	No	No	No
WF_ORDER_EXP	Yes	Yes	No	Yes
WF_COST_BUILDUP_TMPL_HEAD	Yes	No	No	No
WF_COST_BUILDUP_TMPL_DETAIL	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_SUPP_UOM	Yes	No	No	No
V_ITEM_MASTER	Yes	No	No	No
V_STORE	Yes	No	No	No

Design Assumptions

N/A

rplprg (Purge Aged Replenishment Results)

Module Name	rplprg.pc
Description	Purge Aged Replenishment Results
Functional Area	Replenishment
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS316
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The replenishment extraction programs write a number of records to replenishment results. Store orders populate the store orders table. The investment buy process writes records to IB results and the Buyer Worksheet Form populates buyer worksheet manual table. These tables hold information that is relevant to replenishment processes. Over time, records on these tables become unneeded and must be cleared out. The replenishment purge program goes through these tables and clears out those records that are older than a predetermined number of days. The purging cycles (number of days) are maintained as a system parameter.

Restart/Recovery

Because this program performs only deletes, there is no need for restart/recovery or multithreading, and there is no driving cursor. However, this program still needs an entry on restart control to determine the number of records to be deleted between commits.

Key Tables Affected

Table 12–19 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
ALL_TAB_PARTITIONS	Yes	No	No	No
REPL_RESULTS	No	No	No	Yes
BUYER_WKSHT_ MANUAL	No	No	No	Yes
STORE_ORDERS	No	No	No	Yes
IB_RESULTS	No	No	No	Yes

Design Assumptions

N/A

replenishment_purge_job (Purge Aged Replenishment Results)

Module Name	replenishment_purge_job
Description	Purge Aged Replenishment Results
Functional Area	Replenishment
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

The replenishment extraction programs (RPLEXT, REQEXT) write a number of records to REPL_RESULTS. This table holds information that is relevant to replenishment processes. Over time, records on this table become unneeded and must be cleared out.

This background job is composed of one step processing only. It will retain the business logic processing from the original batch program algorithm.

The Business logic program (REPLENISHMENT_PURGE) will invoke a call to a new program specific for handling historical tables such as REPL_RESULTS table that are considered partitioned tables. PARTITION_SQL.PURGE_INTERVAL_PARTITION is called passing the target table name "REPL_RESULTS" and will execute the proper deletion/purging of records from target table by exercising table partitioning handling such as Dropping Interval Partition (same as truncate or delete from table).

The purge program considered the system parameter setting, Replenishment Results Purging Cycle (repl_results_purge_days) to determine those records that are older than a predetermined number of days.

Scheduling Constraints

Table 12–20 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 6 hours interval - 2 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	NA

Restart/Recovery

NA

Key Tables Affected

Table 12–21 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
ALL_TAB_PARTITIONS	Yes	No	No‘	No
ALL_PART_TABLES	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
REPL_RESULTS	No	No	No	Yes

Design Assumptions

NA

rplathistprg (Purge Replenishment Attribute History)

Module Name	rplathistprg.pc
Description	Purge Replenishment Attribute History
Functional Area	Replenishment
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS312
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The batch will purge data from the replenishment attributes update history table if it's older than the defined number of retention weeks as specified in the system parameters.

Restart/Recovery

N/A

Key Tables Affected

Table 12–22 Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_ATTR_UPD_HIST	No	No	No	Yes
SYSTEM_OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
ALL_TAB_PARTITIONS	Yes	No	No	No

Design Assumptions

N/A

rplprg_month (Purge Replenishment Results History by Month)

Module Name	rplprg_month.pc
Description	Purge Replenishment Results History by Month

Functional Area	Replenishment
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS317
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The replenishment extraction programs write a number of records to replenishment results. Store orders populate the store orders table. The investment buy process writes records to IB results and the Buyer Worksheet Form populates buyer worksheet manual table. These tables hold information that is relevant to replenishment processes. Over time, records on these tables become unneeded and must be cleared out.

The monthly replenishment purge program goes through these tables and clears out those records that are older than a predetermined number of days defined as a system parameter. The eways ewInvAdjustToRMS, ewReceiptToRMS need to be shutdown when this program is run.

Restart/Recovery

Because this program performs only deletes, there is no need for restart/recovery or multithreading, and there is no driving cursor. However, this program still needs an entry on restart control to determine the number of records to be deleted between commits.

Key Tables Affected

Table 12–23 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
ALL_TAB_PARTITIONS	Yes	No	No	No
REPL_RESULTS	No	No	No	Yes
BUYER_WKSHT_MANUAL	No	No	No	Yes
STORE_ORDERS	No	No	No	Yes
IB_RESULTS	No	No	No	Yes

Design Assumptions

N/A

repl_indctn_purge.ksh (Purge Scheduled Replenishment Induction Staging Tables)

Module Name	repl_indctn_purge.ksh
Description	Purge scheduled replenishment induction staging tables
Functional Area	Inventory Management
Module Type	Admin
Module Technology	Shell Script
Catalog ID	N/A
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to remove old scheduled replenishment records from the staging tables. Records that are candidates for deletion are:

- Processes that have successfully been processed or processed with warnings that have been uploaded to Merchandising or downloaded to S9T
- Processes in error status where all other related records containing the process ID have been processed successfully
- Processes that are past the data retention days (that is, the action date is earlier than the retention date)

Restart/Recovery

Restart ability will be implied, because the records that are selected from the cursor will be deleted before the commit.

Key Tables Affected

Table 12–24 Key Tables Affected

Table	Select	Insert	Update	Delete
PROC_DATA_ RETENTION_DAYS	Yes	No	No	No
SVC_PROCESS_TRACKER	Yes	No	No	Yes
SVC_REPL_ATTR_UPDATE	Yes	No	No	Yes
CORESVC_PREPL_ERR	Yes	No	No	Yes
S9T_ERRORS	Yes	No	No	Yes
CORESVC_REPL_CHUNKS	Yes	No	No	Yes
S9T_FOLDER	Yes	No	No	Yes
SVC_PROCESS_REPL	Yes	No	No	Yes

Design Assumptions

N/A

replindbatch.ksh (Upload Replenishment Induction Data)

Module Name	replindbatch.ksh
Description	Upload replenishment schedule
Functional Area	Inventory Movement
Module Type	Integration
Module Technology	Ksh
Catalog ID	N/A
Wrapper Script	rmswrap_shell_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to Bulk upload xml file data from template files to a staging table (into the content XML column).

This batch will be responsible for validating the input parameters, below are the list of validations.

- The input file should exist.
- The input file's extension must be ".xml".
- The template_name should be valid. A package function will be called for validation.

Once xml data is loaded into the staging table, the script will do the following:

- Initialize a row in the process tracker table for asynchronous processing.
- Call the main induction process that uploads data into the staging tables, validates and inserts data into the base Merchandising replenishment schedule tables.

Restart/Recovery

N/A

Key Tables Affected

Table 12–25 Key Tables Affected

Table	Select	Insert	Update	Delete
S9T_FOLDER	No	Yes	No	No
S9T_TEMPLATE	Yes	No	No	No

Table 12–25 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_PROCESS_TRACKER	No	Yes	No	No
SVC_REPL_ATTR_UPDATE	Yes	Yes	No	Yes
REPL_ATTR_UPDATE_HEAD	Yes	Yes	Yes	Yes
REPL_ATTR_UPDATE_ITEM	Yes	Yes	Yes	Yes
REPL_ATTR_UPDATE_LOC	Yes	Yes	Yes	Yes
CORESVC_REPL_ERR	Yes	Yes	No	No
S9T_ERRORS	Yes	Yes	No	No

Design Assumptions

N/A

buyer_wksht_purge_job (Purge Aged Buyer Worksheet Results)

Module Name buyer_wksht_purge_job

Description Purge Aged Buyer Worksheet Results

Functional Area Replenishment

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (BUYER_WKSHT_PURGE_THREAD) will filter eligible records from buyer worksheet manual results (BUYER_WKSHT_MANUAL) table based on its purge criteria from system parameter settings. The Replenishment Result Purging Cycle (repl_results_purge_days) parameter will determine those unneeded records that are older than predetermined number of days from its creation date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_BUYER_WKSHT_PURGE_STG.

The Business logic program (BUYER_WKSHT_PURGE) will process all records from the staging table. Using bulk processing, this program will delete old records from buyer worksheet manual (BUYER_WKSHT_MANUAL) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 12–26 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 6 hours interval - 2 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 12–27 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_BUYER_WKSHT_PURGE_STG	Yes	Yes	No	Yes
BUYER_WKSHT_MANUAL	No	No	No	Yes

Design Assumptions

NA

investment_buy_purge_job (Purge Aged Investment Buy Results)

Module Name investment_buy_purge_job

Description Purge Aged Investment Buy Results

Functional Area Replenishment

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (INVESTMENT_BUY_PURGE_THREAD) will filter eligible records from investment buy results (IB_RESULTS) table based on its purge criteria from system parameter settings. The Replenishment Result Purging Cycle (repl_results_purge_days) parameter will determine those unneeded records that are older than predetermined number of days from its creation date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_INVESTMENT_BUY_PURGE_STG.

The Business logic program (INVESTMENT_BUY_PURGE) will process all records from the staging table. Using bulk processing, this program will delete old records from investment buy results (IB_RESULTS) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 12–28 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 6 hours interval - 2 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 12–29 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_INVESTMENT_BUY_PURGE_STG	Yes	Yes	No	Yes
IB_RESULTS	No	No	No	Yes

Design Assumptions

NA

store_orders_purge_job (Purge Aged Store Orders Results)

Module Name	store_orders_purge_job
Description	Purge Aged Store Orders Results
Functional Area	Replenishment
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (STORE_ORDERS_PURGE_THREAD) will filter eligible records from store orders results (STORE_ORDERS) table based on its purge criteria from system parameter settings. The Replenishment Result Purging Cycle (repl_results_purge_days) parameter will determine those unneeded records that are older than predetermined number of days from its creation date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_STORE_ORDERS_PURGE_STG.

The Business logic program (STORE_ORDERS_PURGE) will process all records from the staging table. Using bulk processing, this program will delete old records from store orders results (STORE_ORDERS) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 12–30 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 6 hours interval - 2 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 12–31 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_STORE_ORDERS_PURGE_STG	Yes	Yes	No	Yes
STORE_ORDERS	No	No	No	Yes

Design Assumptions

NA

13

Inventory

Most inventory process is in Merchandising are performed via the UI and near real time RIB integrations. However, some inventory related batch processes exist to manage inventory data.

Batch Design Summary

The following batch designs are included in this chapter:

- edidlprd.pc (Download Sales and Stock On Hand From Merchandising to Suppliers)
- ordinvupld.pc (Upload and Process Inventory Reservations from Sales Audit)
- wasteadj.pc (Adjust Inventory for Wastage Items)
- refeodinventory.ksh (Refresh End of Day Inventory Snapshot)
- invaprg.pc (Purge Aged Inventory Adjustments)
- inv_adj_purge_job (Purge Aged Inventory Adjustments)
- customer_order_purge.ksh (Purge Aged Customer Orders)
- customer_orders_purge_job (Purge Aged Customer Orders)

edidlprd (Download Sales and Stock On Hand From RMS to Suppliers)

Module Name	edidlprd.pc
Description	Download Sales and Stock On Hand From Merchandising to Suppliers
Functional Area	Inventory
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS47
Wrapper Script	rmswrap_multi_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is used to transmit item level sales and stock on hand information to vendors. The report is a summary that will be sent to specified suppliers via EDI giving sales details, as well as current stock on hand and in transit for all locations for each of the items supplied by that supplier. Only those suppliers which have an EDI sales reporting frequency of either daily or weekly will have files generated by this program. The system parameter EDI Daily Report Lag is used for suppliers receiving daily updates to determine the day lag for sales data sent, to account for late posting sales.

Restart/Recovery

Restart/recovery in this program is achieved through utilizing a global temporary table. Once a supplier is processed, it is deleted from the temporary table to prevent the same supplier from being processed again during recovery.

Key Tables Affected

Table 13–1 Key Tables Affected

Table	Select	Insert	Update	Delete
SUPS	Yes	No	No	No
EDI_SUPS_TEMP	Yes	No	No	Yes
EDI_DAILY_SALES	Yes	Yes	Yes	No
PERIOD	Yes	No	No	No
COMPHEAD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_SUPPLIER	Yes	No	No	No
ITEM_LOC_HIST	Yes	No	No	No
ITEM_LOC_SOH_EOD	Yes	No	No	No
ITEM_SUPP_COUNTRY_LOC	Yes	No	No	No

I/O Specification

Integration Type Download from Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000013

Output File Layout

Table 13–2 edidlprd.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File record descriptor	Char(5)	FHEAD	Describes record type
	Line number	Number(10)	0000000001	Sequential file line number
	File source	Char(5)	DLPRD	File Type
	File create date	Char(8)	N/A	Date that the file was created in YYYYMMD D format
THEAD	File record descriptor	Char(5)	THEAD	Identifies record type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
	Report date	Char(8)	N/A	For weekly reporting, this will contain the current date. For daily reporting, it will be the date represented by the sales, current date – lag days. Both will be in the YYYYMMD D format
	Supplier	Number(10)	N/A	Merchandisi ng Supplier Number

Table 13–2 (Cont.) edidlprd.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
TITM	File record descriptor	Char(5)	TITM	Identifies file record type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
	Item	Char(25)	N/A	Transaction level item to which with the data is related
	Item_Num_Type	Char(6)	N/A	Contains the item number type for the item on ITEM_MASTER
	Ref_Item	Char(25)	N/A	Contains the primary reference item for the item in the file, if defined
	Ref_Item_Num_Type	Char(6)	N/A	Contains the item number type for the reference item from ITEM_MASTER
	Vendor catalog number	Char(30)	N/A	Contains the VPN (Vendor Product Number), if defined for the item/ supplier
	Item description	Char(250)	N/A	Contains the transaction level item description from ITEM_MASTER

Table 13–2 (Cont.) edidlprd.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
TQUTY	File record descriptor	Char(5)	TQUTY	Identifies record type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction number	Number(10)	N/A	Sequential transaction number
	Quantity descriptor	Char(15)	N/A	Indicates what the quantity represents, either 'On-hand' (stock), 'Sold'(sales), or 'In transit'
	Location type	Char(2)	N/A	Indicates the type of location represented in the file: 'ST' for store or 'WH' warehouse
	Location	Number(10)	N/A	Contains the store or warehouse number for which the information applies
	Unit cost	Number(20)	N/A	Contains the current unit cost for the item/location with 4 implied decimal places. This value will be in the supplier's currency
	Quantity	Number(12)	N/A	Indicates the quantity of the item sold, on hand or in transit to the location; the quantity is represented with 4 implied decimal places

Table 13–2 (Cont.) edidlprd.pc - Output File

Record Name	Field Name	Field Type	Default Value	Description
TTAIL	File record descriptor	Char(5)	TTAIL	Identifies record type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction lines	Number(6)	N/A	Number of lines for this transaction
FTAIL	File record descriptor	Char(5)	TTAIL	Identifies record type
	Line number	Number(10)	N/A	Total number of lines in file
	Number of transaction lines	Number(10)	N/A	Number of transaction lines in file

Design Assumptions

- A data translator will be used to convert the flat file produced by Merchandising to the required EDI data format.
- Only data for items where the supplier is indicated as the primary supplier/origin country for the item will be included in the report.

ordinvupld (Upload and Process Inventory Reservations from ReSA)

Module Name	ordinvupld.pc
Description	Upload and Process Inventory Reservations from Sales Audit
Functional Area	RMS
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS113
Wrapper Script	batch_ordinvupld.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program processes the input file generated by the Sales Audit Inventory Export batch, which is generated to reserve and un-reserve inventory based on in-store customer orders and layaway. An in-store customer order is one where the customer is purchasing inventory present in the store, but will not take it home immediately. For example, with a large item like a sofa, the customer may pickup at a later time with a larger vehicle. Layaway is when a customer pays for an item over time and only takes the item home once it has been fully paid for. In processing this file,

Merchandising updates the quantity of the item/location sent to either add or subtract from the quantity in the Customer Order inventory status type.

Restart/Recovery

The logical unit of work for this batch program is a valid item status transaction at a given store/location. The logical unit of work is defined as a group of these transaction records. The Oracle Retail standard file-based restart/recovery logic is used. Records are committed to the database when the maximum commit counter is reached.

Key Tables Affected

Table 13–3 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_LOC_SOH	No	No	Yes	No
TRAN_DATA	No	Yes	No	No
ITEM_STATUS_QTY	Yes	Yes	Yes	No
ITEM_MASTER	Yes	No	No	No
STORE	Yes	No	No	No

I/O Specification

Integration Type Upload to Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000049

Input File Layout

Table 13–4 ordinvupld.pc - Input File

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Identifies the file record type
	File Line Id	Char(10)	0000000001	Sequential file line number
	File type definition	Char(4)	ORIN	Identifies the file type
	File Create Date	Char(14)	N/A	File Create Date in YYYYMMDDHHMMSS format
	Location	Number(10)	N/A	Store location number

Table 13–4 (Cont.) ordinvupld.pc - Input File

Record Name	Field Name	Field Type	Default Value	Description
THEAD	Record descriptor	Char(5)	THEAD	Identifies the file record type
	File Line Id	Char(10)	N/A	Sequential file line number
	Transaction Date & Time	Char(14)	Transaction Date	Date and time of the order processed
	Transaction Type	Char(6)	'SALE'	Transaction type code specifies whether the transaction is sale or Return
TDETL	Record descriptor	Char(5)	TDETL	Identifies the file record type
	File Line Id	Char(10)	N/A	Sequential file line number
	Item Type	Char(3)	REF or	Can be REF or ITM
	Item	Char(25)	ITM	Id number of the ITM or REF

Table 13–4 (Cont.) ordinvupld.pc - Input File

Record Name	Field Name	Field Type	Default Value	Description
	Item Status	Char(6)	LIN - Layaway Initiate LCA - Layaway Cancel LCO - Layaway Complete PVLCO - Post void of Layaway complete ORI - Pickup/delivery Initiate ORC - Pickup/delivery Cancel ORD - Pickup/delivery Complete PVORD - Post void of Pick-up/delivery complete	Type of transaction
	Dept	Number(4)	N/A	Department of item sold or returned
	Class	Number(4)	N/A	Class of item sold or returned.
	Sub class	Number(4)	N/A	Subclass of item sold or returned
	Pack Ind	Char(1)	N/A	Pack indicator of item sold or returned
	Quantity Sign	Chanr(1)	'P' or 'N'	Sign of the quantity.
	Quantity	Number(12)	N/A	Quantity * 10000 (4 implied decimal places), number of units for the given order (item) status

Table 13–4 (Cont.) ordinvupld.pc - Input File

Record Name	Field Name	Field Type	Default Value	Description
	Selling UOM	Char(4)	N/A	UOM at which this item was sold
	Catchweight Ind	Char(1)	N/A	Indicates if the item is a catchweight item. Valid values are Y or NULL
	Customer Order number	Char(48)	N/A	Customer Order number
TTAIL	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type
	File Line Identifier	Number(10)	Specified by Sales Audit	ID of current line being processed by input file.
	Transaction count	Number(6)	Specified by Sales Audit	Number of TDETL records in this transaction set
FTAIL	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Identifier	Number(10)	Specified by external system	ID of current line being processed by input file
	File Record Counter	Number(10)	N/A	Number of records/transactions processed in current file (only records between FHEAD & FTAIL)

Design Assumptions

N/A

wasteadj (Adjust Inventory for Wastage Items)

Module Name wasteadj.pc

Description Adjust Inventory for Wastage Items

Functional Area	Inventory
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS388
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program reduces inventory of spoilage type wastage items to account for natural wastage that occurs over the shelf life of the product. This program affects only items with spoilage type wastage identified on ITEM_MASTER with a waste_type of 'SP' (spoilage). Sales type wastage is accounted for at the time of sale.

This program should be scheduled to run prior to the stock count and stock ledger batch to ensure that the stock adjustment taken during the current day is credited to the appropriate day.

Restart/Recovery

The logical unit of work is an item/location. This batch program commits when the number of records processed has reached commit_max_ctr. If the program aborts, it restarts from the last successfully processed item /location.

Design Assumptions

N/A

refeodinventory (Refresh End of Day Inventory Snapshot)

Module Name	refeodinventory.ksh
Description	Refresh End of Day Inventory Snapshot
Functional Area	Inventory Tracking
Module Type	Business Processing
Module Technology	Ksh
Catalog ID	RMS303
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script refreshes the item/location stock on hand end of day snapshot. This end of day snapshot is used for assorted history build programs. The script does the following:

- Truncates the item/location stock on hand end of day snapshot table.
- Inserts all records from the item/location stock on hand table into the item/location stock on hand end of day snapshot table.

Restart/Recovery

N/A

Key Tables Affected

Table 13–5 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_LOC_SOH	Yes	No	No	No
ITEM_LOC_SOH_EOD	No	Yes	No	Yes
SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

- All of the daily updates pertaining to stock on hand have been performed during prior batch phases.
- The executing schema has DROP ANY TABLE privileges. This is needed to perform the truncate on the item/location end of day snapshot table.
- The item/location end of day snapshot table is owned by the schema name specified in the TABLE_OWNER column of the SYSTEM_OPTIONS view.

invaprg (Purge Aged Inventory Adjustments)

Module Name	invaprg.pc
Description	Purge Aged Inventory Adjustments
Functional Area	Inventory
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS251
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program will delete all inventory adjustment records whose adjustment date has elapsed a pre-determined number of months. The number of months that inventory adjustment records are kept before they are purged by this batch is defined by the system parameter Inventory Adjustment Months.

Restart/Recovery

N/A

Key Tables Affected

Table 13–6 Key Tables Affected

Table	Select	Insert	Update	Delete
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
INV_ADJ	No	No	No	Yes

Design Assumptions

N/A

inv_adj_purge_job (Purge Aged Inventory Adjustments)

Module Name inv_adj_purge_adj

Description Purge Aged Inventory Adjustments

Functional Area Inventory

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (INV_ADJ_PURGE_THREAD) will filter eligible records from inventory adjustment (INV_ADJ) table based on its purge criteria from system parameter settings. The Inventory Adjustment Months (inv_adj_months) parameter will determine records to kept before they are purged whole adjustment date has elapsed from a pre-determined number of months. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_INV_ADJ_PURGE_STG.

The Business logic program (INV_ADJ_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from inventory adjustment (INV_ADJ) table. It will free up and clean the staging table

afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 13–7 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 13–8 Key Tables Affected

Table	Select	Insert	Update	Delete
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_INV_ADJ_PURGE_ STG	Yes	Yes	No	Yes
INV_ADJ	No	No	No	Yes

Design Assumptions

NA

customer_order_purge.ksh (Purge Aged Customer Orders)

Module Name customer_order_purge.ksh

Description Purge Aged Customer Orders

Functional Area Purchase Orders

Module Type Admin

Module Technology ksh

Catalog ID RMS205

Wrapper Script rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module will serve as the wrapper for the package function which will purge the store fulfillment customer order records from the customer order tables based on the history months system parameter. This will also purge the obsolete records having the status 'X' where the customer order could not be created.

Restart/Recovery

N/A

Key Tables Affected

Table 13–9 Key Tables Affected

Table	Select	Insert	Update	Delete
ORDCUST	Yes	No	No	Yes
ORDCUST_DETAIL	Yes	No	No	Yes
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No

Design Assumptions

N/A

customer_orders_purge_job (Purge Aged Customer Orders)

Module Name customer_orders_purge_job

Description Download Sales and Stock On Hand From RMS to Suppliers

Functional Area Inventory

Module Type Integration

Module Technology ProC

Catalog ID RMS47

Runtime Parameters NA

Design Overview

This background job is composed of one step processing only. It will retain the business logic processing from original KSH script algorithm.

The Business logic program (CUSTOMER_RESERVE_SQL.PURGE_CUSTOMER_ORDER) will removed all store fulfillment customer order records from purchase customer order (ORDCUST, ORDCUST_DETAIL) tables based on the purge criteria from the system parameter setting, CUST_ORDER_HISTORY_MONTHS from PURGE_CONFIG_OPTIONS table. This will also purge the obsolete records having status 'X' where the customer order could not be created.

Scheduling Constraints

Table 13–10 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 1 hour interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	NA

Restart/Recovery

NA

Key Tables Affected

Table 13–11 Key Tables Affected

Table	Select	Insert	Update	Delete
ORDCUST	Yes	No	No	Yes
ORDCUST_DETAIL	Yes	No	No	Yes
ORDCUST_CUSTOMER_DETAIL	Yes	No	No	No
PURGE_CONFIG_OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No

Security Considerations

NA

14

Transfers, Allocation, and RTV

Transfers, Allocations and Return to Vendor (RTV) transactions move inventory among locations. The majority of processing associated with these transactions occurs through the user interface and near real time RIB integration with Oracle Retail Store Inventory Management (SIM) and Oracle Retail Warehouse Management System (RWMS). However, Merchandising does use a variety of batch programs to maintain the data related to these transactions.

Batch Design Summary

The following batch designs are included in this chapter:

- docclose.pc - Close Transactions with no Expected Appointments, Shipments or Receipts
- doc_queue_close_job - Close Transactions with no Expected Appointments, Shipments or Receipts
- dummyctn.pc - Reconcile Received Dummy Carton IDs with Expected Cartons
- tamperctn.pc - Detail Receive Damaged or Tampered with Cartons
- distropcpub.pc - Stage Regular Price Changes on Open Allocations/Transfers so Publishing Sends New Retail to Subscribing Applications
- mrt.pc - Create Transfers for Mass Return Transfer
- mrtrtv.pc - Create Return To Vendor for Mass Return Transfer
- mrtupd.pc - Close Mass Return Transfers
- mrtprg.pc - Purge Aged Mass Return Transfers and RTVs
- mrt_purge_job - Purge Aged Mass Return Transfers and RTV
- rtvprg.pc - Purge Aged Returns to Vendors
- rtv_purge_job - Purge Aged Returns to Vendors
- tsfclose.pc - Close Overdue Transfers
- transfer_close_job - Close Overdue Transfers
- tsfprg.pc - Purge Aged Transfers
- transfer_purge_job - Purge Aged Transfers
- allocbt.ksh - Create Book Transfers for Allocations Between Warehouses in the Same Physical Warehouse.

docclose (Close Transactions with no Expected Appointments, Shipments or Receipts)

Module Name	docclose.pc
Description	Close Transactions with no Expected Appointments, Shipments or Receipts
Functional Area	Transfers, Allocation, and RTVs
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS219
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will be used to attempt to close POs, transfers, and allocations that do not have any outstanding appointments, shipments or receipts expected. Receipts without appointments are recorded on a queue table. Allocations sourced from an inbound receipt of another document (such as, POs, Transfers, Allocations, ASNs and BOL) can only be closed if the sourcing document is closed. This batch program will retrieve unique documents from the table and use existing functions to attempt closure for each.

Restart/Recovery

The logical unit of work is a unique doc and doc_type combination. The program is restartable on the doc number

Design Assumptions

N/A

doc_queue_close_job (Close Transactions with no Expected Appointments, Shipments or Receipts)

Module Name	doc_queue_close_job
Description	Close Transactions with no Expected Appointments, Shipments or Receipts
Functional Area	Transfers, Allocation, and RTVs
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (DOC_QUEUE_CLOSE_THREAD) will filter eligible records from document close queue (doc_close_queue) table based on un-"C"losed status of POs, Transfers and Allocations. These unique documents that do not habe any outstanding appointments, shipments or receipts expected, receipts without appointments will be recorded on the DOC_CLOSE_QUEUE table. Likewise, allocations sourced from an inbound receipt of another document (eg. POs, Transfers, Allocations, ASNs and BOLs) can only be closed if the sourcing document is closed. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_DOC_CLOSE_QUEUE_STG.

The Business logic program (DOC_QUEUE_CLOSE) will process all records from the staging table. Using bulk processing, this program will update the records of each document type to attempt closure of each document. PO-type documents will call APPT_DOC_CLOSE_SQL.CLOSE_PO program. Transfer-type documents will call APPT_DOC_CLOSE_SQL.CLOSE_TSF program. Allocation-type documents will call APPT_DOC_CLOSE_SQL.CLOSE_ALL_ALLOCS program. All successful closure of document will be removed its entry from DOC_CLOSE_QUEUE table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 14–1 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 1 hour interval - 12 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Document ID/number

Restart/Recovery

NA

Key Tables Affected

Table 14–2 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No

Table 14–2 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
B8D_DOC_CLOSE_QUEUE_STG	Yes	Yes	No	Yes
DOC_CLOSE_QUEUE	Yes	No	No	Yes
ORDHEAD	No	No	Yes	No
DEAL_CALC_QUEUE	No	No	No	Yes
ITEM_LOC_SOH	No	No	Yes	No
TSFHEAD	No	No	Yes	No
ALLOC_HEADER	No	No	Yes	No

Design Assumptions

NA

dummyctn (Reconcile Received Dummy Carton IDs with Expected Cartons)

Module Name	dummyctn.pc
Description	Reconcile Received Dummy Carton IDs with Expected Cartons
Functional Area	Transfers, Allocations and RTVs
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS233
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

When stock orders are received, if a carton number or barcode cannot be read due to damage to the box or other factors, a dummy ID is assigned to it and its detail received at the store or warehouse. The dummy ID and the details of the carton received are then written to a staging table during the receiving process. This batch process scans stock orders to find transfers or allocations that contain cartons that were not received to see if any shipments contain un-received cartons that match the dummy carton receipt (both item and quantity). If a match is found, then the dummy carton is received against the matching carton. If a match is not found, an error is written to an error file and the record remains on the staging table.

Restart/Recovery

This program deletes from the dummy carton staging table. The program will restart by processing the records that remain on the dummy carton staging table.

Key Tables Affected

Table 14–3 Key Tables Affected

Table	Select	Insert	Update	Delete
SHIPSKU_TEMP	Yes	Yes	No	Yes
SHIPMENT	Yes	No	Yes	No
SHIPSKU	Yes	No	Yes	No
PACKITEM	Yes	No	No	No
DUMMY_CARTON_STAGE	Yes	Yes	Yes	Yes
TSFHEAD	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	No
IF_ERRORS	No	Yes	No	No
ALLOC_DETAIL	No	No	Yes	No
SHIPITEM_INV_FLOW	Yes	No	Yes	No
APPT_DETAIL	No	No	Yes	No
DOC_CLOSE_QUEUE	No	Yes	No	No
TRAN_DATA	No	Yes	No	No
ITEM_LOC_SOH	No	Yes	Yes	No
EDI_DAILY_SALES	No	No	Yes	No
STAKE_SKU_LOC	No	Yes	Yes	No
STAKE_PROD_LOC	No	No	Yes	No
MRT_ITEM_LOC	No	No	Yes	No
TSFDETAIL	No	Yes	Yes	No
NWP	No	Yes	Yes	No
INV_ADJ	No	Yes	No	No
TSFDETAIL_CHRG	No	Yes	No	No
ITEM_LOC	No	Yes	No	No
PRICE_HIST	No	Yes	No	No
ITEM_SUPP_COUNTRY_LOC	No	Yes	Yes	No
ITEM_SUPP_COUNTRY_BRACKET_COST	No	Yes	Yes	No
INV_STATUS_QTY	No	Yes	Yes	Yes

Design Assumptions

N/A

tamperctn (Detail Receive Damaged or Tampered with Cartons)

Module Name tamperctn.pc

Description Detail Receive Damaged or Tampered with Cartons

Functional Area	Transfers, Allocations and RTVs
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS371
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program looks for items that were intended to be received as a pack and attempts to match based on component quantity. It reads records from the staging table for the carton ID for pack items not received and attempts to match on the components of the pack and quantity. If a match is found, then the dummy carton is received against the matching carton. If a match is not found, an error is written to an error file and the record remains on the staging table.

This program is only run if the Receive Pack Component system parameter is set to Yes.

Restart/Recovery

N/A

Key Tables Affected

Table 14–4 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
DUMMY_CARTON_STAGE	Yes	No	No	Yes
PERIOD	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	No
SHIPMENT	Yes	No	No	No
SHIPSKU	Yes	No	No	No
SHIPSKU_TEMP	Yes	Yes	No	Yes
PACKITEM	Yes	No	No	No

Design Assumptions

N/A

distropcpub (Stage Regular Price Changes on Open Allocations/Transfers so Publishing Sends New Retail to Subscribing Applications)

Module Name	distropcpub.pc

Description	Stage Regular Price Changes on Open Allocations/Transfers so Publishing Sends New Retail to Subscribing Applications
Functional Area	Transfers, Allocations, and RTV
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS216
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will look for any regular price change (transaction type 4 or 11 from the price history table) that is due to go into effect tomorrow. Then, for any open allocations or transfers where the 'to' location and items that have price changes going into effect, it places a record on the allocation or transfer publishing queue tables, such that they can be picked up by the RIB and sent to the subscribing systems.

Restart/Recovery

The logical unit of work is store. The driving cursor retrieves all item/locations that have price changes in effect from the next day. It also gets all of the component items of the non-sellable packs that have price changes.

Key Tables Affected

Table 14–5 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
PRICE_HIST	Yes	No	No	No
V_RESTART_STORE	Yes	No	No	No
V_PACKSKU_QTY	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No
TSFHEAD	Yes	No	No	No
TSDETAIL	Yes	No	No	No
ORDHEAD_REV	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ALLOC_MFQUEUE	No	Yes	No	No
TSF_MFQUEUE	No	Yes	No	No

I/O Specification

Integration Type	Download from Merchandising
File Name	N/A
Integration Contract	IntCon000196 ALLOC_MFQUEUE table

Integration Type	Download from Merchandising
File Name	N/A
Integration Contract	IntCon000197 TSF_MFQUEUE table

Design Assumptions

N/A

mrt (Create Transfers for Mass Return Transfer)

Module Name	mrt.pc
Description	Create Transfers for Mass Return Transfer
Functional Area	Transfers, Allocations and RTVs
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS273
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program creates individual transfers for each 'from' location on an approved Mass Return Transfer. Transfers will be created in approved status, however for MRTs with a Quantity Type of 'Manual', meaning the MRT was created for a specific quantity rather than 'All Inventory', if the SOH at the sending location is lower than the requested quantity the status will be created in Input status. In addition, if the Transfer Not After Date specified on the MRT is earlier than or equal to the current date, the status of the associated transfers will also be set to Input.

Restart/Recovery

The logical unit of work is a from/to location combination. This may represent a transfer of multiple items from a location (store or warehouse) to a warehouse, depending on how the MRT was created. Restart/recovery is based on from/to location as well. The batch program uses the restart all locations view to thread processing by warehouse (to location).

Key Tables Affected

Table 14–6 Key Tables Affected

Table	Select	Insert	Update	Delete
MRT	Yes	No	Yes	No
MRT_ITEM	Yes	No	No	No
MRT_ITEM_LOC	Yes	No	Yes	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	Yes	No
TSFDETAIL	Yes	Yes	Yes	No
TSFHEAD	No	Yes	Yes	No
TRAN_DATA	No	Yes	No	No
INV_STATUS_QTY	Yes	Yes	Yes	Yes
PERIOD	Yes	No	No	No

Design Assumptions

N/A

mrtrtv (Create Return to Vendor for Mass Return Transfer)

Module Name	mrtrtv.pc
Description	Create Return To Vendor for Mass Return Transfer
Functional Area	Transfers, Allocations and RTVs
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS275
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program creates RTVs for approved mass return transfers that require an RTV to be created automatically and have an RTV create date earlier than or equal to the current date. RTVs are created in either Input or Approved status, depending on how the MRT was created. The program will then set the status of all processed MRTs to 'R' in the MRT table, which indicates that the RTVs have been created.

Restart/Recovery

The logical unit of work for this program is set at the warehouse level. Threading is done by store using the restart all locations view.

Key Tables Affected

Table 14–7 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
MRT	Yes	No	Yes	No
MRT_ITEM	Yes	No	No	No
MRT_ITEM_LOC	Yes	No	No	No
SUPS	Yes	No	No	No
RTV_HEAD	No	Yes	Yes	No
RTV_DETAIL	No	Yes	No	No
ADDR	Yes	No	No	No

Design Assumptions

N/A

mrtupd (Close Mass Return Transfers)

Module Name	mrtupd.pc
Description	Close Mass Return Transfers
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS276
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program updates the status of MRTs and their associated transfers to closed status, for MRTs or transfers associated with an MRT that remain open after the transfer and/or RTV not after dates have passed. MRTs that have transfers in progress (shipped but not received) will not be closed by this program.

Restart/Recovery

The logical unit of work for this program is warehouse. This program is multi-threaded using the restart all locations view.

Key Tables Affected

Table 14–8 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
MRT	Yes	No	Yes	No
TSFHEAD	Yes	No	Yes	No
SHIPSKU	Yes	No	No	No
TSFDETAIL	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	Yes	No
V_PACKSKU_QTY	Yes	No	No	No

Design Assumptions

N/A

mrtprg (Purge Aged Mass Return Transfers and RTV)

Module Name	mrtprg.pc
Description	Purge Aged Mass Return Transfers and RTVs
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS274
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to purge mass return transfer (MRT) records, and their associated transfers and RTVs. Only MRTs with a status of closed in which all transfers associated with the MRT are also closed and where the time elapsed between the current date and the close date is at least equal to the system parameter value for MRT Transfer Retention days.

Restart/Recovery

The logical unit of work for this batch program is a warehouse location. The program is multithreaded using restart all locations view.

Key Tables Affected

Table 14–9 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
TSFHEAD	Yes	No	No	Yes
TSFDETAIL	No	No	No	Yes
SHIPMENT	No	No	No	Yes
SHIPSKU	Yes	No	No	Yes
SHIPITEM_INV_FLOW	No	No	No	Yes
CARTON	No	No	No	Yes
APPT_HEAD	Yes	No	No	Yes
APPT_DETAIL	Yes	No	No	Yes
DOC_CLOSE_QUEUE	No	No	No	Yes
INVC_HEAD	Yes	No	No	Yes
INVC_DETAIL	Yes	No	No	Yes
MRT	Yes	No	No	Yes
MRT_ITEM	Yes	No	No	Yes
MRT_ITEM_LOC	Yes	No	No	Yes
RTV_HEAD	Yes	No	No	Yes
RTV_DETAIL	No	No	No	Yes
TSFDETAIL_CHRG	No	No	No	Yes

Design Assumptions

N/A

mrt_purge_job (Purge Aged Mass Return Transfers and RTV)

Module Name	mrt_purge_job
Description	Purge Aged Mass Return Transfers and RTVs
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (MRT_PURGE_THREAD) will filter eligible records from mass return transfer (MRT) table based on its purge criteria from system parameter settings. The MRT Transfer Retention days (tsf_mrt_retention_days) parameter will determine the time elapsed with MRT close date is less than the current date. Only MRTs with closed status (and all associated transfers that are also closed) are captured for deletion. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_MRT_PURGE_STG.

The Business logic program (MRT_PURGE_THREAD) will process all records from the staging table. Using bulk processing, this program will purge the records from mass return transfer (MRT) table and its associated transfers and RTVs. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 14–10 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by warehouse

Restart/Recovery

NA

Key Tables Affected

Table 14–11 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	Yes
B8D_MRT_PURGE_STG	Yes	Yes	No	Yes
TSFHEAD	Yes	No	No	Yes
TSFDETAIL	No	No	No	Yes
SHIPMENT	No	No	No	Yes
SHIPSKU	Yes	No	No	Yes
SHIPITEM_INV_FLOW	No	No	No	Yes

Table 14–11 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
CARTON	No	No	No	Yes
APPT_HEAD	Yes	No	No	Yes
APPT_DETAIL	Yes	No	No	Yes
DOC_CLOSE_QUEUE	No	No	No	Yes
INVC_HEAD	Yes	No	No	Yes
INVC_DETAIL	Yes	No	No	Yes
MRT	Yes	No	No	Yes
MRT_ITEM	Yes	No	No	Yes
MRT_ITEM_LOC	Yes	No	No	Yes
RTV_HEAD	Yes	No	No	Yes
RTV_DETAIL	No	No	No	Yes
TSFDETAIL_CHRG	No	No	No	Yes

Design Assumptions

NA

rtvprg (Purge Aged Returns to Vendors)

Module Name	rtvprg.pc
Description	Purge Aged Returns to Vendors
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS320
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program purges outdated RTV transactions from Merchandising. RTVs are considered outdated if they number of months between their completion date and the current date exceeds the system parameter RTV Order History Months and where all debit memos associated with the RTV have been posted.

Restart/Recovery

N/A

Key Tables Affected

Table 14–12 Key Tables Affected

Table	Select	Insert	Update	Delete
RTV_HEAD	No	No	No	Yes
RTV_DETAIL	No	No	No	Yes
INVC_HEAD	Yes	No	No	Yes
INVC_DETAIL	No	No	No	Yes
INVC_NON_MERCH	Yes	No	No	Yes
INVC_MERCH_VAT	Yes	No	No	Yes
INVC_DETAIL_VAT	Yes	No	No	Yes
INVC_MATCH_QUEUE	Yes	No	No	Yes
INVC_DISCOUNT	Yes	No	No	Yes
INVC_TOLERANCE	Yes	No	No	Yes
ORDLOC_INVC_COST	Yes	No	Yes	No
INVC_MATCH_WKSHT	Yes	No	No	Yes
INVC_XREF	Yes	No	No	Yes
RTVITEM_INV_FLOW	No	No	No	Yes
RTV_HEAD_CFA_EXT	No	No	No	Yes

Design Assumptions

N/A

rtv_purge_job (Purge Aged Returns to Vendors)

Module Name rtv_purge_job

Description Purge Aged Returns to Vendors

Functional Area Transfers, Allocations and RTVs

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (RTV_PURGE_THREAD) will filter eligible records from return-to-vendor header (RTV_HEAD) table based on its purge criteria from system parameter settings. The RTV Order History Months (rtv_order_history_months) parameter will determine the number of months between their completion date and current date exceeds as defined to be outdated records. These old/outdated RTV records should have all debit memos associated to have been posted. These records

are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_RTV_PURGE_STG.

The Business logic program (RTV_PURGE_THREAD) will process all records from the staging table. Using bulk processing, this program will delete the records from return-to-vendor header (RTV_HEAD) and other related/associated RTV tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 14–13 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by RTV Order Number

Restart/Recovery

NA

Key Tables Affected

Table 14–14 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_RTV_PURGE_STG	Yes	Yes	No	Yes
RTV_HEAD	No	No	No	Yes
RTV_DETAIL	No	No	No	Yes
INVC_HEAD	Yes	No	No	Yes
INVC_DETAIL	No	No	No	Yes
INVC_NON_MERCH	Yes	No	No	Yes
INVC_MERCH_VAT	Yes	No	No	Yes
INVC_DETAIL_VAT	Yes	No	No	Yes
INVC_MATCH_QUEUE	Yes	No	No	Yes
INVC_DISCOUNT	Yes	No	No	Yes
INVC_TOLERANCE	Yes	No	No	Yes

Table 14–14 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ORDLOC_INVC_COST	Yes	No	Yes	No
INVC_MATCH_WKSHT	Yes	No	No	Yes
INVC_XREF	Yes	No	No	Yes
RTVITEM_INV_FLOW	No	No	No	Yes
RTV_HEAD_CFA_EXT	No	No	No	Yes

Design Assumptions

NA

tsfclose (Close Overdue Transfers)

Module Name	tsfclose.pc
Description	Close Overdue Transfers
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS379
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program processes unshipped and partially shipped transfers that are considered 'overdue', based on system parameter settings. If this functionality is enabled, then this program will evaluate transfers to determine if they are overdue. The way that a transfer is considered overdue depends on the source and destination locations. There are separate system parameters for each of store to store, store to warehouse, warehouse to store, and warehouse to warehouse types of transfers.

For unshipped transfers, the transfer status is updated to delete and transfer reserved and expected inventory is backed out from the table for the sending and receiving locations respectively. For transfers that are shipped but not fully received, an entry is made into the document close queue table. These transfers are picked up by docclose batch and closed after reconciliation.

Restart/Recovery

The logical unit of work for this module is defined as a unique tsf_no. The restart transfer view is used for threading. This batch program uses table-based restart/recovery. The commit happens in the database when the commit max counter is reached.

Key Tables Affected

Table 14–15 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
INV_MOVE_UNIT_ OPTIONS	Yes	No	No	No
TSFHEAD	Yes	No	Yes	No
ALLOC_HEADER	Yes	No	Yes	No
ITEM_MASTER	Yes	No	No	No
V_PACKSKU_QTY	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	Yes	No
DOC_CLOSE_QUEUE	No	Yes	No	No

Design Assumptions

N/A

transfer_close_job (Close Overdue Transfers)

Module Name transfer_close_job

Description Close Overdue Transfers

Functional Area Transfers, Allocations and RTVs

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (TRANSFER_CLOSE_THREAD) will filter eligible records from transfer header (TSFHEAD) table and other associated transfer-related tables. Based on its system parameter settings, it will process unshipped and partitally shipped 'overdue' transfers. If this functionality is enabled (by setting the system parameter TSF_CLOSE_OVERDUE = 'Y'), then this program will evaluate transfers to determine if they are overdue. The way that a transfer is considered overdue depends on the source and destination locations. There are separate system parameters for each of store to store, store to warehouse, warehouse to store, and warehouse to warehouse types of transfers. . These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_ TRANSFER_CLOSE_STG.

The Business logic program (TRANSFER_CLOSE) will process all records from the staging table. Using bulk processing, this program will update the records from transfer header and other associated transfer tables. For unshipped transfers, the transfer status is updated to delete and transfer reserved and expected inventory is

backed out on ITEM_LOC_SOH for the sending and receiving locations respectively. For transfers that are shipped but not fully received, an entry is made into doc_close_ queue table. These transfers are picked up by docclose batch and closed after reconciliation. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 14–16 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 1 hour interval - 12 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Transfer number

Restart/Recovery

NA

Key Tables Affected

Table 14–17 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
INV_MOVE_UNIT_ OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_TRANSFER_CLOSE_ STG	YEs	Yes	No	Yes
TSFHEAD	Yes	No	Yes	No
ALLOC_HEADER	Yes	No	Yes	No
ITEM_MASTER	Yes	No	No	No
V_PACKSKU_QTY	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	Yes	No
DOC_CLOSE_QUEUE	No	Yes	No	No

Design Assumptions

NA

tsfprg (Purge Aged Transfers)

Module Name	tsfprg.pc
Description	Purge Aged Transfers
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS380
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module purges closed or deleted transfers and their associated records after a set number of days, based on the Transfer History Months system parameter.

Restart/Recovery

This batch program is multithreaded using the restart transfer view. The logical unit of work is a transfer number. This batch program commits to the database for every commit max counter number of transfers processed.

Key Tables Affected

Table 14–18 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
TSFHEAD	Yes	No	No	Yes
TSFDETAIL	No	No	No	Yes
ALLOC_HEADER	Yes	No	No	Yes
ALLOC_DETAIL	No	No	No	Yes
ALLOC_CHRG	Yes	No	No	Yes
ALLOC_PURGE_QUEUE	Yes	No	No	No
DOC_PURGE_QUEUE	Yes	No	No	No
SHIPSKU	Yes	No	No	Yes
CARTON	No	No	No	Yes

Design Assumptions

- This batch program does not process Mass Return Transfers (MRT) and Franchise transfers (FO and FR). Purging of MRT and Franchise Order and Return records are done by mrtprg, wfordprg, wfrtnprg respectively.

transfer_purge_job (Purge Aged Transfers)

Module Name	transfer_purge_job
Description	Purge Aged Transfers
Functional Area	Transfers, Allocations and RTVs
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (TRANSFER_PURGE_THREAD) will filter eligible records from transfer header (TSFHEAD) table and other associated transfer-related tables based on its purge criteria from system parameter settings. The Transfer History Months (tsf_history_mths) parameter will determine what transfer records are ready for purging that are considered closed or deleted in status and comparison of transfer close date with tomorrow's current date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_TRANSFER_PURGE_STG.

The Business logic program (TRANSFER_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from transfer header and other associated transfer tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 14–19 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by transfer number

Restart/Recovery

NA

Key Tables Affected

Table 14–20 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_TRANSFER_PURGE_STG	Yes	Yes	No	Yes
TSFHEAD	Yes	No	No	Yes
TSFDETAIL	No	No	No	Yes
ALLOC_HEADER	Yes	No	No	Yes
ALLOC_DETAIL	No	No	No	Yes
ALLOC_CHRG	Yes	No	No	Yes
ALLOC_PURGE_QUEUE	Yes	No	No	No
DOC_PURGE_QUEUE	Yes	No	No	No
SHIPSKU	Yes	No	No	Yes
CARTON	No	No	No	Yes
TSFHEAD_CFA_EXT	No	No	No	Yes
TSFHEAD_L10N_EXT	No	No	No	Yes
TSF_ITEM_COST	No	No	No	Yes
TSF_ITEM_WO_COST	No	No	No	Yes
TSF_PACKING	Yes	No	No	Yes
TSF_PACKING_DETAIL	No	No	No	Yes
TSF_XFORM	Yes	No	No	Yes
TSF_XFORM_DETAIL	No	No	No	Yes
TSF_WO_HEAD	Yes	No	No	Yes
TSF_WO_DETAIL	No	No	No	Yes
TSFDETAIL_CHRG	No	No	No	Yes
ORDCUST	Yes	No	No	Yes
ORDCUST_CUSTOMER_DETAIL	No	No	No	Yes
ORDCUST_DETAIL	No	No	No	Yes
ORDCUST_PUB_INFO	No	No	No	Yes
SHIPITEM_INV_FLOW	No	No	No	Yes
SHIPSKU_PRG_HIST	No	Yes	No	No
ORDCUST_DETAIL_PRG_HIST	No	Yes	No	No

Table 14–20 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ORDCUST_PRG_HIST	No	Yes	No	No
TSFDETAIL_CHRG_PRG_HIST	No	Yes	No	No
TSF_WO_DETAIL_PRG_HIST	No	Yes	No	No
TSF_WO_HEAD_PRG_HIST	No	Yes	No	No
TSF_XFORM_DETAIL_PRG_HIST	No	Yes	No	No
TSF_XFORM_PRG_HIST	No	Yes	No	No
TSF_PACKING_DETAIL_PRG_HIST	No	Yes	No	No
TSF_PACKING_PRG_HIST	No	Yes	No	No
TSF_ITEM_WO_COST_PRG_HIST	No	Yes	No	No
TSF_ITEM_COST_PRG_HIST	No	Yes	No	No
TSFDETAIL_PRG_HIST	No	Yes	No	No
TSFHEAD_L10N_EXT_PRG_HIST	No	Yes	No	No
TSFHEAD_CFA_EXT_PRG_HIST	No	Yes	No	No
TSFHEAD_PRG_HIST	No	Yes	No	No

Design Assumptions

NA

allocbt (Create Book Transfers for Allocations Between Warehouses in the Same Physical Warehouse)

Module Name	allocbt.ksh
Description	Create Book Transfers for Allocations Between Warehouses in the Same Physical Warehouse
Functional Area	Inventory Movement
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS175
Wrapper Scripts	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In Merchandising, when an allocation is received that involves a movement of stock between two warehouses, it should be determined if the source and any of the destination warehouses belong to the same physical warehouse. If so, that portion of the allocation should be treated as a book transfer and not sent down to RWMS for processing. This batch job identifies such allocations and creates book transfers once the allocation source is received and/or the release date for the allocation is reached.

Allocations can be sourced either from a warehouse's available inventory or from an inbound receipt. These allocations are integrated into Merchandising's Allocation tables and can be identified as the following:

1. Warehouse Sourced Allocations:
    a. Order number is NULL and doc is NULL on the allocation header table.
2. Purchase Ordered Sourced Allocations (Cross Doc POs):
    a. Order number holds the PO number and doc type is 'PO' on the allocation header table.
    b. Linked shipments are identified through the order number value on the shipment and the allocation header tables.
3. Transfer Sourced Allocations:
    a. Order number holds the transfer number and doc type is 'TSF' on the allocation header table.
    b. Linked shipments are identified through the distro number on the shipment/item table and the order number on the allocation header table.
4. Allocation Sourced from an Inbound Allocation:
    a. Order number holds the allocation number and doc type is 'ALLOC' on the allocation header table.
    b. Linked shipments are identified through the distribution number on the shipment/item table and the order number on the allocation header table.
5. ASN Sourced Allocations:
    a. Doc holds the ASN number and doc type is 'ASN' on the allocation header table.
    b. Linked shipments are identified through the ASN on the shipment table and the doc on the allocation header table.
6. BOL Sourced Allocations:
    a. Doc holds the BOL number and doc type is 'BOL' on the allocation header table.
    b. Linked shipments are identified through the BOL number on the shipment table and the doc value on the allocation header table.

This batch job supports all above allocation scenarios and calls a core package function to create book transfers.

Restart/Recovery

N/A

Key Tables Affected

Table 14–21 Key Tables Affected

Table	Select	Insert	Update	Delete
ALLOC_HEADER	Yes	No	Yes	No
ALLOC_DETAIL	Yes	No	Yes	No
ITEM_LOC_SOH	Yes	No	Yes	No
WH	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
TSFHEAD	Yes	No	No	No
TSFDETAIL	Yes	No	Yes	No
SHIPMENT	Yes	No	No	No
SHIPSKU	Yes	No	No	No

Design Assumptions

N/A

15

Sales Posting

Merchandising includes a convenient interface with your point-of-sale system (POS) that allows you to efficiently upload sales transaction data. Once the data enters Merchandising, other modules take over the posting of that data to sales transaction, sales history, and stock-on-hand tables. This overview describes the upload and validation of sales transaction data from your POS to Merchandising and the relevant processes.

Creating a POSU File

The Merchandising Sales Posting module, uploadsales.ksh requires a POSU file that is rolled up to the item/store/price point level. There are a variety of ways to create this file:

- If you use Oracle Retail Xstore Point of Service (ORPOS), the integration through Sales Audit will create appropriate POSU files.
- If you integrate your POS and Sales Audit, packaged integration between Sales Audit and Merchandising will produce POSU files.
- If you integrate your OMS (Order Management System) and Sales Audit, packaged integration between Sales Audit and Merchandising will produce POSU files.
- If you use a 3rd party POS or Order Management System (OMS) and do not use Sales Audit, you must use a custom process to roll up data to an item/store/price point level
- Additional information about the structure of the POSU file is available in the detailed discussion of the uploadsales.ksh process.

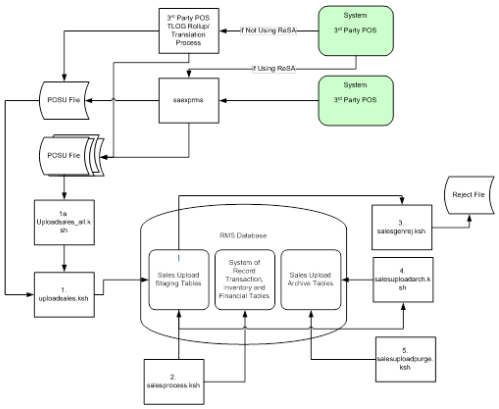
Sales Posting Business Process

The Sales Posting Process consists of a number of related programs.

1. uploadsales.ksh reads the POSU file and writes it's contents to a series of staging tables.
    a. uploadsales_all.ksh wraps uploadsales.ksh to simplify the process of running uploadsales.ksh for groups of POSU files.
2. I thireads the staged data and performs major validation, financial and inventory processing. Details of this processing are below in the detailed discussion of salesprocess.ksh.

3. salesgenrej.ksh creates a reject file for transactions that fail salesprocess.ksh validation.
4. salesuploadarch.ksh archives successfully processed transactions and clears them out of the staging tables.
5. salesuploadpurge.ksh purges transactions from the archive tables after the transactions age out of the system.

Figure 15–1 Sales Posting Business Process



Batch Design Summary

The following batch designs are included in this chapter

- uploadsales.ksh (Upload POSU File for Processing)
- uploadsales_all.ksh (Process Multiple POSU Files)
- salesprocess.ksh (Main Processing of Staged Sale/Return Transactions)
- salesgenrej.ksh (Reject POSU Transactions)
- salesuploadarch.ksh (Archive Successfully Posted Transactions)
- salesuploadpurge.ksh (Purge Aged Archived POSU Transactions)
- sales_reprocess.ksh (Re-processing of Sale/Return Transactions Due to Chunk Not Process Issue)
- file_upload_errors_purge.ksh (Purge FILE_UPLOAD_STATUS and FILE_ UPLOAD_ERRORS Tables)

uploadsales.ksh (Upload POSU File for Processing)

Module Name	uploadsales.ksh
Description	Upload POSU File for Processing
Functional Area	Sales Posting
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS112

Wrapper Script batch_uploadsales.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to upload the contents of the POSU file from Sales Audit or 3rd Party POS to the staging table for further processing.

Restart/Recovery

N/A

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

The number of threads, the amount of waiting time, number for retries, and average volume of data should be considered. RETRY_WAIT_TIME shouldn't be increased significantly.

The rows, bindsize and readsize parameter of the sqlldr command can be configured for better performance. This gives more control over how many times the inserts are committed/executed.

Security Considerations

N/A

I/O Specification

Integration Type Upload to Merchandising

File Name POSU_<store>_<tran_date>_<sysdate>.<thread_val>

Integration Contract IntCon000044

Input File Layout

Table 15–1 Input File

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type

Table 15–1 (Cont.) Input File

Record Name	Field Name	Field Type	Default Value	Description
	File Line Identifier	Number(10)	Specified by external system	ID of current line being processed by input file
	File Type Definition	Char(4)	POSU	Identifies file as 'POS Upload'
	File Create Date	Char(14)	N/A	Date file was written by external system
	Location Number	Number(10)	N/A	Store identifier
	Vat include indicator	Char(1)	N/A	Determines whether or not the store stores values including vat. Not required but populated by Sales Audit
	Vat region	Number(4)	N/A	Vat region the given location is in. Not required but populated by Sales Audit
	Currency code	Char(3)	N/A	Currency of the given location. Not required but populated by sales audit
	Currency retail decimals	Number(1)	N/A	Number of decimals supported by given currency for retails. Not required but populated by Sales Audit
Transaction Header	File Type Record Descriptor	Char(5)	THEAD	Identifies transaction record type
	File Line Identifier	Number(10)	Specified by external system	ID of current line being processed by input file
	Transaction Date	Char(14)	Transaction date	Date sale/return transaction was processed at the POS
	Item Type	Char(3)	REF or ITM	Item type will be represented as a REF or ITM
	Item Value	Char(25)	N/A	The ID number of an ITM or REF
	Dept	Number(4)	N/A	Dept of item sold or returned. Not required but populated by Oracle Retail Sales Audit
	Class	Number(4)	N/A	Class of item sold or returned. Not required but populated by Oracle Retail Sales Audit
	Subclass	Number(4)	N/A	Subclass of item sold or returned. Not required but populated by Oracle Retail Sales Audit

Table 15–1 (Cont.) Input File

Record Name	Field Name	Field Type	Default Value	Description
	Pack Indicator	Char(1)	N/A	Pack indicator of item sold or returned. Not required but populated by Oracle Retail Sales Audit
	Item level	Number(1)	N/A	Item level of item sold or returned. Not required but populated by Oracle Retail Sales Audit
	Tran level	Number(1)	N/A	Tran level of item sold or returned. Not required but populated by Oracle Retail Sales Audit
	Wastage Type	Char(6)	N/A	Wastage type of item sold or returned. Not required but populated by Oracle Retail Sales Audit
	Wastage Percent	Number(12)	N/A	Wastage Percent*10000 (4 implied decimal places.), wastage percent of item sold or returned. Not required but populated by Oracle Retail Sales Audit
	Transaction Type	Char(1)	S - sales R - return	Transaction type code to specify whether transaction is a sale or a return
	Drop Shipment Indicator	Char(1)	Y N	Indicates whether the transaction is a drop shipment or not. If it is a drop shipment, indicator will be 'Y'. This field is not required, but will be defaulted to 'N' if blank
	Total Sales Quantity	Number(12)	N/A	Total sales quantity * 10000 (4 implied decimal places), number of units sold at a particular location
	Selling UOM	Char(4)	N/A	UOM at which this item was sold
	Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative
	Total Sales Value	Number(20)	N/A	Total Sales Value * 10000 (4 implied decimal places), sales value, net sales value of goods sold
	Last Modified Date	Char(14)	N/A	For VBO future use
	Catchweight Indicator	Char(1)	NULL	Indicates if the item is a catch weight item. Valid values are 'Y' or NULL

Table 15–1 (Cont.) Input File

Record Name	Field Name	Field Type	Default Value	Description
	Actual Weight Quantity	Number(12)	NULL	Actual Weight Quantity*10000 (4 implied decimal places), the actual weight of the item, only populated if catchweight_ ind = 'Y'
	Sub Trantype Indicator	Char(1)	NULL	Tran type for Sales Audit Valid values are 'A', 'D', NULL
	Total Igtax Value	Number(20)	N/A	Total Igtax Value * 10000 (4 implied decimal places), goods sold or returned
	Sales Type	Char(1)	N/A	Indicates whether the line item is a Regular Sale, a customer order serviced by OMS (External CO) or a customer order serviced by a store (In Store CO). Valid values are 'R','E', or 'I'
	No Inventory Return Indicator	Char(1)	N/A	Contains an indicator that identifies a return without inventory. This is generally a non-required column, but in case of Returns, this is required. Valid values are 'Y' or 'N'
	Return Disposition	Char(10)	N/A	Contains the disposition code published by RWMS as part of the returns upload to OMS
	Return Warehouse	Number(10)	N/A	Contains the physical warehouse ID for the warehouse identifier where the item was returned
	Customer Order No	Char(48)	N/A	This column contains the customer order number ID.
	Fulfillment Order No	Char(48)	N/A	This column contains the fulfillment order number ID.
	Fulfillment Loc Type	Char(2)	N/A	This column contains the fulfillment location type. Code for the fulfillment loc type from code_detail where code_type = 'FLTP'
	Fulfillment Loc	Number(10)	N/A	This column contains the fulfillment loc ID.
	Orig Store	Number(10)	N/A	This column contains the original store value for a Return transaction.

Table 15–1 (Cont.) Input File

Record Name	Field Name	Field Type	Default Value	Description
	POS Tran Id	Number(20)	N/A	This column contains the unique identifier for a sale transaction. This is an **Optional** field. Customer needs to provide a unique identifier for a sale transaction. If not provided, Merchandising assigns a unique value from a DB sequence.
Transaction Tax	File Type Record Descriptor	Char(5)	TTAX	Identifies the file record type
	File Line Identifier	Number(10)	Specified by external system	Sequential file line number
	Tax Code	Char(6)	N/A	Holds the tax code associated to the item
	Tax Rate	Number(20)	N/A	Tax rate*10000000000(10 implied decimal places), holds the tax rate for the tax code associated to the item
	Total Tax Value	Number(20)	N/A	Total Tax value*10000(4 implied decimal places), total tax amount for the line item
Transaction Detail	File Type Record Descriptor	Char(5)	TDETL	Identifies transaction record type
	File Line Identifier	Number(10)	Specified by external system	ID of current line being processed by input file
	Promotional Tran Type	Char(6)	N/A	Code for promotional type from code_detail, code_type = 'PRMT'
	Promotion Number	Number(10)	N/A	Promotion number from the Merchandising
	Sales Quantity	Number(12)	N/A	Sales quantity*10000 (4 implied decimal places.), number of units sold in this prom type
	Sales Value	Number(20)	N/A	Sales value*10000 (4 implied decimal places.), value of units sold in this prom type
	Discount Value	Number(20)	NA	Discount quantity*10000 (4 implied decimal places.), value of discount given in this prom type
	Promotion Component	Number(10)	N/A	Links the promotion to additional pricing attributes

Table 15–1 (Cont.) Input File

Record Name	Field Name	Field Type	Default Value	Description
Transaction Trailer	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type
	File Line Identifier	Number(10)	Specified by external system	ID of current line being processed by input file
	Transaction Count	Number(6)	Specified by external system	Number of TDETL records in this transaction set
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Identifier	Number(10)	Specified by external system	ID of current line being processed by input file
	File Record Counter	Number(10)	N/A	Number of records/transactions processed in current file (only records between fhead & ftail)

Fields expected in POSU format based on changes adopted:

	V16	V16+ Customer Order Changes	V19
Fulfillment Order No	No	Yes	Yes
Fulfillment Loc Type	No	Yes	Yes
Fulfillment Loc	No	Yes	Yes
Orig Store	No	Yes	Yes
POS Tran Id	No	No	Yes

Design Assumptions

Multiple taxes for an item if sent from POS to Sales Audit, will be summed to a single tax in Merchandising and assigned one of the applicable tax codes.

Rolling up transactions to the item/store/price point

The program uploadsales.ksh requires that transactions be rolled up the item/store/price point level. The tables below give a hypothetical (though not particularly realistic) example of the type of rollup required by upload_sales.ksh.

Table 15–2 Sales for Item Number 1234 (at one store during one period of the day)

Transaction Number	Number of Items Sold	Amount (in specified currency unit)	Price point (price reason)
167	1	9.99	Regular
395	2	18.00	Promotional
843	1	7.99	Clearance

Table 15–2 (Cont.) Sales for Item Number 1234 (at one store during one period of the

Transaction Number	Number of Items Sold	Amount (in specified currency unit)	Price point (price reason)
987	3	27.00	Promotional
1041	1	9.99	Regular
1265	4	31.96	Clearance

Note: The variation of the price per item in different transactions. This is the result of the price applied at the time of sale—the price point. Now look at the next table that shows the same transactions rolled up by item and price point.

Table 15–3 Sales for Item Number 1234

Number of Items Sold	Price Reason (price point)	Total Amount for Item-Price point (in currency)
2	Regular price	19.98
5	Promotional price	45.00
5	Clearance price	39.95

uploadsales.ksh takes the totals and looks for any discounts for transactions in the POSU file. It applies the discounts to an expected total dollar amount using the price listed for that item from the pricing table (PRICE_HIST). It next compares this expected total against the reported total. If the program finds a discrepancy between the two amounts, it is reported. If the two totals match, the rollup is considered valid. If value-added tax (VAT) is included in any sales transaction amounts, it is removed from those transactions prior to the validation process.

Reject File

The module produces a reject file similar to the input file if it is found to have missing or duplicate FHEAD or FTAIL records. Records in these types of files are loaded to the svc_posupld_load table, but not in the svc_posupld_staging table.

uploadsales_all.ksh (Process Multiple POSU Files)

Module Name	uploadsales_all.ksh
Description	Process Multiple POSU Files
Functional Area	Sales Posting
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS157
Wrapper Script	batch_uploadsales.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this script is to execute the uploadsales.ksh module for all POSU files that are for upload. This wrapper will simplify the sales upload process for multiple POSU files, removing the need to call the uploadsales.ksh individually for each file.

Restart/Recovery

N/A

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

The number of threads, the amount of waiting time, number for retries, and average volume of data should be considered. RETRY_WAIT_TIME shouldn't be increased significantly.

The rows, bindsize and readsize parameter of the sqlldr command can be configured for better performance. This gives more control over how many times the inserts are committed/executed.

Security Considerations

N/A

I/O Specification

Integration Type	Upload to Merchandising
File Name	POSU_<store>_<tran_date>_<sysdate>.<thread_val>
Integration Contract	IntCon000044

Input File Layout

Refer to the Input File Layout section in uploadsales.doc.

salesprocess.ksh (Main Processing of Staged Sale/Return Transactions)

Module Name	salesprocess.ksh
Description	Main Processing of Staged Sale/Return Transactions
Functional Area	Sales Posting

Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS151
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of the SALESPROCESS.KSH module is to process sales and return details from an external point of sale system (either POS or OMS). The sales/return transactions will be validated against Oracle Retail item/store relations to ensure the sale is valid, but this validation process can be eliminated if the sales that are being passed in, have been screened by sales auditing. The following common functions will be performed on each sales/return record read from the input file:

- Read sales/return transaction record
- Lock associated record in Merchandising
- Validate item sale
- Check whether TAX maintenance is required, and if so determine the TAX amount for the sale.
- Write all financial transactions for the sale and any relevant markdowns to the stock ledger.
- Post item/location/week sales to the relevant sales history tables
- Perform last sales processing to maintain accurate sales information in the system

POSU Chunking

Table 15–4 Concurrent Threads and Chunk Size

MAX_CONCURRENT_THREADS	MAX_CHUNK_SIZE
2	3

Number of Threads: 11

Thread 1	Chunk 1	THEAD 1	Item 1
Thread 1	Chunk 1	THEAD 2	Item 1
Thread 1	Chunk 1	THEAD 3	Item 2
Thread 1	Chunk 1	THEAD 4	Item 2
Thread 1	Chunk 1	THEAD 5	Item 3
Thread 2	Chunk 2	THEAD 6	Item 5
Thread 2	Chunk 2	THEAD 7	Item 6
Thread 2	Chunk 2	THEAD 8	Item 7

Thread 3	Chunk 3	THEAD 9	Item 8
Thread 3	Chunk 3	THEAD 10	Item 9
Thread 3	Chunk 3	THEAD 11	Item 10

In this run, threads would be allocated first to chunks 1 and 2. The other threads would only be picked up once either thread 1 or 2 has finished its processing.

Restart/Recovery

The logical unit of work for salesprocess.ksh is a set of a single or multiple valid item sales transactions at a given store location. This set is defined as a chunk. Based on the example above, if for some reason, chunk 2 raised an error, THEAD 4, 5, and 6 wouldn't be posted in Merchandising. Other chunks, if there are no errors, would be processed. User has to correct the transaction details and upload the updated POSU file that includes the affected THEAD lines for reprocessing.

Locking Strategy

Since the sales upload processes are run multiple times a day in a trickle-polling system, a locking mechanism is put in place to allow on-line transactions and the salesprocess.ksh module to run at the same time.

Because multithreading logic based on chunks is applied, it is possible that a record is locked by another thread. Without a mechanism to perform waiting/retrying, record locking errors would happen more frequently.

In the table RMS_PLSQL_BATCH_CONFIG, RETRY_LOCK_ATTEMPTS is the number of times the thread will try to acquire the lock for a table and RETRY_WAIT_TIME is the number of seconds the thread will wait before it retries. Once the number of retries is equal to the limit defined, the whole chunk wouldn't be processed. This would create a reject file with which you can use to upload again to the staging table.

Security Considerations

N/A

Performance Considerations

The number of threads, the amount of waiting time, number for retries, and average volume of data should be considered.

Be careful when increasing the number of threads. When the number exceeds the capacity of the server, new jobs wouldn't be able to start when this program is running and would impact other users of the system.

Because this is multithreaded and can be executed during the store day, it is prone to locking errors. Record locking errors would happen if the thread reached the maximum number of retries (RETRY_LOCK_ATTEMPT) to fetch the lock. To prevent this, increase the value of the retries and let the value of RETRY_WAIT_TIME remain at 1. This means that it would retry every second until the maximum number of retries have been reached.

It is also important to know the average volume of data. It is a determinant of what would be the chunk size. If the chunk is too small, it couldn't utilize processing the records in bulk. If the chunk size is too large, in such that, all records would be in one chunk, it wouldn't utilize the multithreaded approach and thus, be inefficient.

I/O Specification

Integration Type	Upload to Merchandising
File Name	N/A; at this point, the POSU data has already been uploaded to the staging tables
Integration Contract	IntCon000103

The module will have the ability to re-process a POSU reject file directly. The file format will therefore be identical to the input file layout for the uploadsales.ksh process. A reject line counter will be kept in the program and is required to ensure that the file line count in the trailer record matches the number of rejected records. If no errors occur, no reject files would be generated.

Design Assumptions

Tax Handling:

POS can send either transactional level tax details in TTAX lines or item-level tax details in IGTAX lines through the RTLOG file to Sales Audit. These tax details will be passed on to Merchandising in the TTAX lines of the POSU file. Even though POS can pass multiple IGTAX/TTAX lines to Sales Audit and from Sales Audit to Merchandising, Merchandising only supports one tax code per item. If multiple taxes for an item are sent from POS to Sales Audit, they will be summed to a single tax in Merchandising sales upload process and assigned one of the applicable tax codes when writing tran_data 88.

Financial Transactions

salesprocess.ksh writes transaction records to the TRAN_DATA table primarily through its write_tran_data function. From the entire list of valid transaction codes (For the full list of transaction codes, see the chapter "General ledger batch" in this volume of the Merchandising Operations Guide), for the column TRAN_CODE, salesupload.ksh writes the following:

Table 15–5 Transaction Records

Transaction Code	Description
01	Net Sales (retail & cost)
02	Net sales (retail & cost) where - retail is always VAT exclusive, written only if system_options.stkldgr_vat_incl_retl_ind = Y
03	Non-inventory Items Sales/Returns
04	Customer Returns (retail & cost)
05	Non-inventory VAT Exclusive Sales
06	Deal Income (sales)
11	Markup (retail only)
12	Markup cancel (retail only)
13	Permanent Markdown (retail only)
14	Markdown cancel (retail only)
15	Promotional Markdown (retail only), including 'in-store' markdown

Table 15–5 (Cont.) Transaction Records

Transaction Code	Description
20	Purchases (retail & cost)
24	Return to Vendor (RTV) from inventory (retail & cost)
60	Employee discount (retail only)

Note: Where value-added-tax is enabled (system_options table, stkldgr_vat_incl_retl_ind column shows 'Y') and the retail accounting method is also enabled, salesupload.ksh writes an additional transaction record for code 02.

Any items sold on consignment are written as a code 20 (Purchases) as well as a 01 (Net Sales) along with all other applicable transactions, like returns. The 20 reflects the fact that the item is purchased at the time it is sold, in other words, a consignment sale.

salesgenrej.ksh (Reject POSU Transactions)

Module Name	salesgenrej.ksh
Description	Reject POSU Transactions
Functional Area	Sales Posting
Module Type	Business Processing
Module Technology	KSH
Catalog ID	RMS338
Wrapper Script	batch_salesgenrej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to archive the rejected transactions and create a reject file based on the recently processed POSU file which is still in the staging table.

Restart/Recovery

N/A

Performance Considerations

The number of threads, the amount of waiting time, number for retries, and average volume of data should be considered. RETRY_WAIT_TIME shouldn't be increased significantly.

Reject File:

The module will have the ability to re-process the reject file directly. The file format will therefore be identical to the input file layout. A reject line counter will be kept in the program and is required to ensure that the file line count in the trailer record matches the number of rejected records. If no errors occur, no reject files would be generated.

salesuploadarch.ksh (Archive Successfully Posted Transactions)

Module Name	salesuploadarch.ksh
Description	Archive Successfully Posted Transactions
Functional Area	Sales Processing
Module Type	Admin
Module Technology	Ksh
Catalog ID	RMS340
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to archive the successfully posted transactions, and clear the staging table.

Performance Considerations

Since the archive tables would be handling a large volume of data. Administrators should consider enlarging the tablespace to accommodate the average volume of data.

Design Assumptions

N/A

salesuploadpurge.ksh (Purge Aged Archived POSU Transactions)

Module Name	salesuploadpurge.ksh
Description	Purge Aged Archived POSU Transactions
Functional Area	Sales Processing
Module Type	Admin
Module Technology	Ksh
Catalog ID	RMS341
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is delete the archive tables for the rejects and the posted transaction based on the given retention period.

Performance Considerations

The retention period for the archived data should be carefully considered. Disregarding this would result in the tablespace size reaching its limit and would not be able to accommodate additional archive records.

Design Assumptions

N/A

sales_reprocess.ksh (Re-processing of Sale/Return Transactions Due to Chunk Not Process Issue)

Module Name	sales_reprocess.ksh
Description	Re-processing of Sale/Return Transactions Due to Chunk Not Process Issue
Functional Area	Sales Posting
Module Type	Business Processing
Module Technology	ksh
Catalog ID	N/A

Design Overview

The purpose of the SALES_REPROCESS.KSH module is to reprocess sales and return details that were marked in error with error chunk not process.

Scheduling Constraints

Table 15–6 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Phase 2 (minimum) Can also be run Ad Hoc
Frequency	Daily
Scheduling Considerations	This program should run right after salesprocess.ksh. It should be run in at least phase 2. Can also be run ad hoc to trickle poll sales.
Pre-Processing	salesprocess.ksh

Table 15–6 (Cont.) Scheduling Constraints

Schedule Information	Description
Post-Processing	salesgenrej.ksh salesuploadarch.ksh
Threading Scheme	Run one Thread; Expecting low volume of POSU files needing reprocessing on chunk not processed issue.

Restart/Recovery

The logical unit of work for sales_reprocess.ksh is a chunk.

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

Key Tables Affected

Table 15–7 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_POSUPLD_LOAD	Yes	Yes	Yes	No
SVC_POSUPLD_LOAD_ ARCH	Yes	No	No	Yes
SVC_POSUPLD_STAGING	Yes	Yes	Yes	No
SVC_POSUPLD_STAGING_ REJ	No	No	No	Yes
SVC_POSUPLD_STATUS	Yes	No	Yes	No

Integration Contract

Integration Type N/A

File Name N/A

Integration Contract N/A

Design Assumptions

N/A

file_upload_errors_purge.ksh (Purge FILE_UPLOAD_STATUS and FILE_UPLOAD_ERRORS Tables)

Module Name	file_upload_errors_purge.ksh
Description	Purge FILE_UPLOAD_STATUS and FILE_UPLOAD_ERRORS Tables.
Functional Area	Administration
Module Type	Admin
Module Technology	Ksh
Catalog ID	N/A
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this program is to purge FILE_UPLOAD_STATUS and FILE_UPLOAD_ERRORS tables regularly in Merchandising.

To validate the status of sales file upload process in Merchandising, the error handling in sales upload process has been enhanced to capture the following attributes of file upload status in FILE_UPLOAD_STATUS and FILE_UPLOAD_ERRORS tables.

- Filename
- Status
- # of lines in file
- # of Records uploaded
- # of Records failed processing
- Date/Time process started
- Date/Time processing completes
- Location (store or warehouse where file originated). For stock counts this would be the physical warehouse.

If errors are identified, the error message, line text and line ID are captured in the FILE_UPLOAD_ERRORS table. The FILE_UPLOAD_STATUS and FILE_UPLOAD_ERRORS tables are replicated thru golden gate, so that customer can verify the upload file results through DAS views.

The file_upload_errors_purge.ksh script is scheduled to run as part of the nightly batch, to purge FILE_UPLOAD_STATUS and FILE_UPLOAD_ERRORS tables regularly in Merchandising based on the retention days input parameter.

Restart/Recovery

This program does not contain restart/recovery logic.

I/O Specification

N/A

Design Assumptions

N/A

16

Sales History

Merchandising maintains sales history at a variety of levels. Item level sales history drives Merchandising replenishment, ratio build and is exported to planning applications (see chapter Integration – Planning in this document). Merchandising also maintains a smoothed average history for Pricing. Sales history rolled up to levels of the merchandise hierarchy is used by Allocation. Many clients also find sales history data useful for custom reporting.

Batch Design Summary

The following batch designs are included in this chapter:

- hstbld.pc (Weekly Sales History Rollup by Department, Class, and Subclass)
- hstbld_diff.pc (Weekly Sales History Rollup by Diff)
- hstbldmth.pc (Monthly Sales History Rollup by Department, Class, and Subclass)
- hstbldmth_diff.pc (Monthly Sales History Rollup by Diffs)
- hstmthupd.pc (Monthly Stock on Hand, Retail and Average Cost Values Update)
- hstwkupd.pc (Weekly Stock on Hand and Retail Value Update for Item/Location)
- hstprg.pc (Purge Aged Sales History)
- history_purge_job (Purge Aged Sales History)
- hstprg_diff.pc (Purge Aged Sales History by Diff)
- hist_diff_purge_job (Purge Aged Sales History by Diff)

hstbld (Weekly Sales History Rollup by Department, Class, and Subclass)

Module Name	hstbld.pc
Description	Weekly Sales History Rollup by Department, Class, and Subclass
Functional Area	Sales History
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS239
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The sales history rollup routine will extract sales history information for each item from the item and item location history tables. The history information will be rolled up to the subclass, class, and dept level to be written to history tables.

The rebuild program can be run in one of two ways:

First, if the program is run with a run-time parameter of 'rebuild', the program will read data (dept, class, and subclass) off the manually input HIST_REBUILD_MASK table, which will determine what to rebuild.

Secondly, if the program is run with a run-time parameter of 'weekly', the program will build sales information for all dept/class/subclass combinations only for the current end of week date.

Restart/Recovery

The logical unit of work for this program is set at the store/dept/class level. Threading is done by store using the v_restart_store view.

The commit_max_ctr field on the RESTART_CONTROL table will determine the number of transactions that equal a logical unit of work.

Design Assumptions

N/A

hstbld_diff (Weekly Sales History Rollup by Diff)

Module Name	hstbld_diff.pc
Description	Weekly Sales History Rollup by Diff
Functional Area	Sales History
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS240
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The sales history rollup routine will extract sales history information for each item_parent from the ITEM_LOC_HIST table. The history information will be rolled up to the item differentiator level to be written to: item_diff_loc_hist and item_parent_loc_hist.

For each item, data to be retrieved includes sales qty and stock. This data must be collected from several tables including ITEM_LOC_HIST, ITEM_LOC, and ITEM_MASTER.

Restart/Recovery

N/A

Design Assumptions

N/A

Key Tables Affected

Table 16–1 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_PARENT_LOC_HIST	No	Yes	Yes	No
ITEM_DIFF_LOC_HIST	No	Yes	Yes	No
ITEM_LOC	Yes	No	No	No
ITEM_LOC_HIST	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
SYSTEM_VARIABLES	Yes	No	No	No
PERIOD	Yes	No	No	No

hstbldmth (Monthly Sales History Rollup By Department, Class And Subclass)

Module Name	hstbldmth.pc
Description	Monthly Sales History Rollup by Department, Class, and Subclass
Functional Area	Sales History
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS241
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The monthly sales history roll up routine will extract sales history information for each item from the ITEM_MASTER and ITEM_LOC_HIST_MTH (item location history by month) tables. The history information will be rolled up to the subclass, class and dept level to be written to: subclass_sales_hist_mth (subclass/location/month/sales type),

class_sales_hist_mth (class/location/month/sales type) and dept_sales_hist_mth (department/location/month/sales type).

This program may be run in parallel with hstbld since they both read from HIST_REBUILD_MASK. The table HIST_REBUILD_MASK table must not be truncated before both programs finish running.

Restart/Recovery

The logical unit of work for the hstbldmth module is department, location, sales type and end of month date with a recommended commit counter setting of 1,000. Processed records are committed each time the record counter equals the maximum recommended commit number.

Design Assumptions

N/A

hstbldmth_diff (Monthly Sales History Rollup By Diffs)

Module Name	hstbldmth_diff.pc
Description	Monthly Sales History Rollup by Diffs
Functional Area	Sales History
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS242
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The sales history rollup routine will extract sales history information for each ITEM_PARENT from the ITEM_LOC_HIST_MTH table and rolls the data to month level. The history information will be rolled up to the item differentiator level to be written to: item_diff_loc_hist_mth and item_parentloc_hist_mth. For each item, data to be retrieved includes sales quantity and stock. This data must be collected from several tables including ITEM_LOC_HIST_MTH, ITEM_LOC, and ITEM_MASTER.

Restart/Recovery

N/A

Locking Strategy

The package HSTBLD_DIFF_PROCESS locks the following tables for update:

ITEM_DIFF_LOC_HIST_MTH

ITEM_PARENTLOC_HIST_MTH

Design Assumptions

N/A

hstmthupd (Monthly Stock on Hand, Retail and Average Cost Values Update)

Module Name	hstmthupd.pc
Description	Monthly Stock on Hand, Retail and Average Cost Values Update
Functional Area	Sales History
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS158
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program runs monthly to update the stock on hand, retail values and average cost for each item/location on the ITEM_LOC_HIST_MTH (item location history by month) table. If the item/location does not exist on the ITEM_LOC_HIST_MTH table, then the new record is written to a comma delimited file which is later uploaded to ITEM_LOC_HIST_MTH table using SQL*Loader (hstmthupd.ctl).

Restart/Recovery

The logical unit of work for this program is the item/location record. Threading is done by store using the v_restart_store_wh view. The commit_max_ctr field on the RESTART_CONTROL table will determine the number of transactions that equal a logical unit of work. Table-based restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000175 hstmthupd.ctl

hstwkupd (Weekly Stock on Hand and Retail Value Update for Item/Location)

Module Name	hstwkupd.pc

Description	Weekly Stock on Hand and Retail Value Update for Item/Location
Functional Area	Sales History
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS159
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program runs weekly to update the current stock on hand, retail values and average cost for each item/location on ITEM_LOC_HIST is using SQL*Loader (hstwkupd.ctl). The program must be run on the last day of the week as scheduled.

Restart/Recovery

The logical unit of work for HSTWKUPD is item/location. The program is threaded by location using the v_restart_store_wh view.

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000176 hstwkupd.ctl

Design Assumptions

N/A

hstprg (Purge Aged Sales History)

Module Name	hstprg.pc
Description	Purge Aged Sales History
Functional Area	Sales Posting
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS244
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Deletes records from ITEM_LOC_HIST, SUBCLASS_SALES_HIST, CLASS_SALES_HIST, DEPT_SALES_HIST and DAILY_SALES_DISCOUNT tables, where data is older than the specified number of months. Number of months for retention of fashion style history is specified by system_options.ITEM_HISTORY_MONTHS.

Restart/Recovery

N/A

Design Assumptions

N/A

history_purge_job (Purge Aged Sales History)

Module Name	history_purge_job
Description	Purge Aged Sales History
Functional Area	Sales Posting
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (HIST_PURGE_THREAD) will filter eligible records from department, class, subclass sales history (DEPT_SALES_HIST, CLASS_SALES_HIST, SUBCLASS_SALES_HIST) tables based on its purge criteria from system parameter settings. The Item History Months (item_history_months) parameter will determine record which is older than the specific number of retention months of fashion style history. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_HIST_PURGE_STG.

The Business logic program (HIST_PURGE_THREAD) will process all records from the staging table. Using bulk processing, this program will delete the records from sales history tables by department, class and subclass tables. It will also invoke a call to a new program specific for handling historical tables that are considered partitioned tables. PARTITION_SQL.PURGE_INTERVAL_PARTITION is called passing each target table names "ITEM_LOC_HIST", "ITEM_LOC_HIST_MTH", and "DAILY_SALES_DISCOUNT" This called program will execute the proper deletion/purging of records from target table by exercising table partitioning handling such as Dropping Interval Partition (same as truncate or delete from table). There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 16–2 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad-hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 16–3 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_HIST_PURGE_STG	Yes	Yes	No	Yes
ALL_PART_TABLES	Yes	No	No	No
ALL_TAB_PARTITIONS	Yes	No	No	No
ITEM_LOC_HIST	No	No	No	Yes
ITEM_LOC_HIST_MTH	No	No	No	Yes
SUBCLASS_SALES_HIST	No	No	No	Yes
CLASS_SALES_HIST	No	No	No	Yes
DEPT_SALES_HIST	No	No	No	Yes
DAILY_SALES_DISCOUNT	No	No	No	Yes

hstprg_diff (Purge Aged Sales History by Diff)

Module Name hstprg_diff.pc

Description Purge Aged Sales History by Diff

Functional Area Sales History

Module Type Admin

Module Technology	ProC
Catalog ID	RMS245
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The tables, ITEM_DIFF_LOC_HIST and ITEM_PARENT_LOC_HIST are purged of sales history differentiator data, which is older than a specified system set date. This date is stored in the purge_config_options.ITEM_HISTORY_MONTHS column.

Restart/Recovery

N/A

Design Assumptions

N/A

hist_diff_purge_job (Purge Aged Sales History by Diff)

Module Name	hist_diff_purge_job
Description	Purge Aged Sales History by Diff
Functional Area	Sales History
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (HIST_DIFF_PURGE_THREAD) will filter eligible records from item-parent-location history by diff (ITEM_PARENT_LOC_HIST) and item-location history by diff (ITEM_DIFF_LOC_HIST) tables based on its purge criteria from system parameter settings. The Item History Months (item_history_months) parameter will determine old sales history differentiator data on a specified system set date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_HIST_DIFF_PURGE_STG.

The Business logic program (HIST_DIFF_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from sales history differentiator (ITEM_DIFF_LOC_HIST) and (ITEM_PARENT_LOC_HIST) tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 16–4 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad-hoc
Frequency	Daily - 1 hour interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 16–5 Key Tables Affected

Table	Select	Insert	Update	Delete
PURGE_CONFIG_OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_HIST_DIFF_PURGE_STG	Yes	Yes	No	Yes
ITEM_DIFF_LOC_HIST	No	No	No	Yes
ITEM_PARENT_LOC_HIST	No	No	No	Yes

17

Stock Count

A stock count is a comparison of an inventory snapshot at a point in time to an actual inventory count received from a location. Stock count batch processes can be divided into two rough categories, processes that prepare future stock counts and processes that process results for today's stock counts. The programs stkschedxpld.pc and stkxpld.pc prepare future stock counts. All other programs process results from today's stock counts.

For more information about Stock Counts, including the interaction of UI and batch processes and data flow, see the Oracle Retail Merchandising Functional Library (Doc ID: 1585843.1).

Note: The White Papers in this library are intended only for reference and educational purposes and may not reflect the latest version of Oracle Retail software.

Batch Design Summary

The following batch designs are included in this functional area:

- stkschedxpld.pc (Create Stock Count Requests Based on Schedules)
- stkxpld.pc (Explode Stock Count Requests to Item Level)
- lifstkup.pc (Conversion of RWMS Stock Count Results File to Merchandising Integration Contract)
- stockcountupload.ksh (Upload Stock Count Results from Stores/Warehouses)
- stockcountprocess.ksh (Process Stock Count Results)
- stkprg.pc (Purge Aged Stock Count)
- stock_count_purge_job (Purge Aged Stock Count)
- stkschedxpld (Create Stock Count Requests Based on Schedules)
- stake_sched_explode_job (Create Stock Count Requests Based on Schedules)
- stkupd (Stock Count Snapshot Update)
- stkvar.pc (Update Stock On Hand Based on Stock Count Results)
- stkdly.pc (Calculate Actual Current Shrinkage and Budgeted Shrink to Apply to Stock Ledger)

lifstkup (Conversion of RWMS Stock Count Results File to RMS Integration Contract)

Module Name	lifstkup.pc
Description	Conversion of RWMS Stock Count Results File to Merchandising Integration Contract
Functional Area	Stock Counts
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS150
Wrapper Script	batch_lifstkup.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Stock Upload Conversion batch is used when RWMS sends count information to Merchandising. This batch converts the inventory balance upload file into the format supported by the Stock Count Upload process.

Restart/Recovery

Oracle Retail standard file-based restart/recovery is used. The commit max counter field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O. The recommended commit counter setting is 1000 records (subject to change based on implementation).

Key Tables Affected

Table 17–1 Key Tables Affected

Table	Select	Insert	Update	Delete
WH	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
STAKE_HEAD	Yes	No	No	No
STAKE_LOCATION	Yes	No	No	No

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integratin Contract	IntCon000172 (input from RWMS) IntCon000102 (output for Merchandising stockcountupload)

Input File Layout

Table 17–2 Input File Layout

Field Name	Field Type	Description
DC_DEST_ID	11 – Number (10) + 1 for trailing space	Unique identifier for the warehouse
TRANSACTION_DATE	15 – Date (14) + 1 for trailing space	Date on which the transaction occurred
ITEM_ID	26 - Varchar2 (25) + 1 for trailing space	Uniquely identifies the item on the count
AVAILABLE_QTY	15 – Number (12) + 1 for leading sign and + 1 for decimal and + 1 for trailing space	Units available for distribution
DISTRIBUTED_QTY	14 – Number (12) + 1 for decimal and + 1 for trailing space	Units distributed include: Units distributed but not yet picked, units picked but not yet manifested, units manifested but not yet shipped
RECEIVED_QTY	15 - Number (12) + 1 for leading sign and + 1 for decimal and + 1 for trailing space	Units received but not put away
TOTAL_QTY	14 – Number (12,4) + 1 for decimal and + 1 for trailing space	Sum of all units that physically exist: container status of: I, D, M, R, T, X
AVAILABLE_WEIGHT	15 – Number (12,4) + 1 for leading sign + 1 for decimal + 1 for trailing space	Weight available for distribution of catch weight items
RECEIVED_WEIGHT	14 – Number (12,4) + 1 for decimal + 1 for trailing space	Weight received but not put away for catch weight items
DISTRIBUTED_WEIGHT	14 – Number (12,4) + 1 for decimal + 1 for trailing space	Weight distributed includes: weight distributed but not yet picked, weight picked but not yet manifested, weight manifested but not yet shipped (value only catch weight items)
TOTAL_WEIGHT	13 – Number (12,4) + 1 for decimal	Sum of all weight that physically exist: container status of: I, D, M, R, T, X. For catch weight items

Output File Layout

Table 17–3 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	file type record descriptor	Char (5)	FHEAD	Describes the file line type
	file line identifier	Number (10)	0000000001	ID of current line being processed
	file type	Char (4)	'STKU'	Identifies the file type
	stocktake_ date	Date (14)	N/A	The date on which the count occurred, formatted as YYYYMMDD HH24MISS
	file create date	Date (14)	N/A	Date on which the file was created, formatted as YYYYMMDD HH24MISS
	cycle count	Number (8)	N/A	stake_ head.cycle_ count
	Location type	Char (1)	'W'	Will always be 'W', as this process is only executed for warehouse locations
	location	Number(10)	N/A	Indicates the number of the physical warehouse where the count occurred

Table 17–3 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FDETL	file type record descriptor	Char(5)	FDETL	Identifies the file line type
	file line identifier	Number(10)	N/A	ID of current line being processed, internally incremented
	Item type	Char(3)	'ITM'	Indicates the type of item that was counted. This will always be 'ITM', indicating a transaction level item
	item value	Char(25)	N/A	The ID of the item that was counted
	inventory quantity	Number(12)	N/A	The total quantity or weight of product counted; includes four implied decimal places
	location description	Char(150)	N/A	Used by Merchandising to determine the location where the item was counted. This program will always leave as NULL
FTAIL	file type record descriptor	Char(5)	FTAIL	Identifies the file line type
	file line identifier	Number(10)	N/A	ID of current line being processed, internally incremenated
	file record count	Number(10)	N/A	Indicates the number of detail records

Design Assumptions

N/A

stockcountupload.ksh (Upload Stock Count Results from Stores/Warehouses)

Module Name	stockcountupload.ksh
Description	Upload Stock Count Results from Stores/Warehouses
Functional Area	Stock Count
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS153
Wrapper Script	batch_stockcountupload.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to upload the contents of the stock count file, which contains the results of a count that occurred in a store or warehouse, to staging tables for further processing.

Key Tables Affected

Table 17–4 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_STKUPLD_FHEAD	Yes	Yes	Yes	Yes
SVC_STKUPLD_FDETL	Yes	Yes	Yes	Yes
SVC_STKUPLD_STATUS	Yes	Yes	Yes	Yes
FILE_UPLOAD_STATUS	No	Yes	Yes	No

Input/Out Specification

Integration Type	Upload in Merchandising
File Name	Determined by runtime parameter
Integratin Contract	IntCon000102

Input File Layout

Table 17–5 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File head descriptor	Char(5)	FHEAD	Describes file line type
	File line identifier	Number(10)	0000000001	ID of current line being processed
	File Type	Char(4)	STKU	Identifies the file type
	File create date	Char(14)	N/A	Indicates the date the file was created in YYYYMMDD HH24MISS format
	Stock take date	Char(14)	N/A	Date on which stock count will take place in YYYYMMDD HHMISS format
	Cycle count	Number (8)	N/A	Unique number to identify the stock count
	Location Type	Char(1)	N/A	Indicates the type of location where the count occurred. Valid values are 'S','W','E'.
	Location	Number(10)	N/A	The location where the stock count occurred

Table 17–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
Transaction Record	File record descriptor	Char(5)	FDETL	Describes file line type
	Line Number	Number(10)	N/A	Sequential file line number
	Item type	Char(3)	N/A	Indicates the type of item counted – either transaction level (ITM) or reference item (REF)
	Item value	Char(25)	N/A	Unique identifier for item that was counted
	Inventory quantity	Number(12)	N/A	Total quantity counted for the item at the location formatted with 4 implied decimal places
	Location description	Char(150)	N/A	Description of inventory location (such as,. sales floor, backroom)
FTAIL	File record descriptor	Char(5)	FTAIL	Marks end of file
	File line identifier	Number(10)	N/A	ID of current line being processed, internally incremented
	File record count	Number(10)	N/A	Number of detail records

Design Assumptions

- This program uses grep to search log files for errors. The GREP function should point to the /usr/xpg4/bin/ directory instead of /usr/bin directory to utilize the "-E" option. Otherwise, it will fail with an "illegal option" error message.

stkdly (Calculate Actual Current Shrinkage and Budgeted Shrink to Apply to Stock Ledger)

Module Name stkdly.pc

Description Calculate Actual Current Shrinkage and Budgeted Shrink to Apply to Stock Ledger

Functional Area	Stock Counts
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS359
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Stock Count Shrinkage Update batch calculates the 'value' variances for Unit & Value stock counts. The main functions are to calculate actual shrinkage amount that is used to correct the book stock value on the stock ledger and to calculate a budgeted shrinkage rate that will be applicable until the next count. Additionally, future transaction data snapshots are aggregated and stored into a table which will be used for shrinkage calculations in month end stock ledger batch process. The month end stock ledger batch process then uses these values when calculating ending inventory for the month.

Restart/Recovery

This batch program is multithreaded using the restart department view. The logical unit of work for this program is department/class/location.

Key Tables Affected

Table 17–6 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
SYSTEM_VARIABLES	Yes	No	No	No
STAKE_PROD_LOC	Yes	No	Yes	No
STAKE_HEAD	Yes	No	No	No
DEPS	Yes	No	No	No
HALF_DATA_BUDGET	Yes	No	No	No
DAILY_DATA	Yes	No	No	No
WEEK_DATA	No	No	Yes	No
MONTH_DATA	Yes	No	Yes	No
HALF_DATA	No	No	Yes	No
DAILY_DATA_TEMP	No	Yes	No	No
TRAN_DATA_FUTURE_SNAPSHOT	Yes	No	No	No
MV_LOC_SOB	Yes	No	No	No

Table 17–6 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
TRAN_DATA_FUTURE_AGGREGATE	No	Yes	Yes	No

Design Assumptions

N/A

stkprg (Purge Aged Stock Count)

Module Name	stkprg.pc
Description	Purge Stock Count
Functional Area	Stock Counts
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS360
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Purge Stock Counts is a data cleanup process to remove old counts from Merchandising. This batch process deletes records from the stock count tables with a stock take date earlier than the last end of month start date or those that have been otherwise flagged for delete. This process deletes records from stock count header and all corresponding child tables.

Restart/Recovery

This program is multi-threaded based on location and the logic of restart and recovery is based on cycle count and location. The deletion of stock count header and stock count product tables is performed in prepost as a post action.

This is done because stkprg is multi-threaded and each thread may have only deleted part of cycle count detail records; hence the records from stock count head and stock count product can only be deleted in the post program when all the details have been deleted.

Key Tables Affected

Table 17–7 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_VARIABLES	Yes	No	No	No
STAKE_LOCATION	Yes	No	No	Yes

Table 17–7 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
STAKE_QTY	No	No	No	Yes
STAKE_CONT	No	No	No	Yes
STAKE_SKU_LOC	No	No	No	Yes
STAKE_PROD_LOC	No	No	No	Yes
STAKE_PRODUCT	No	No	No	Yes
STAKE_HEAD	Yes	No	No	Yes

Design Assumption

N/A

stock_count_purge_job (Purge Aged Stock Count)

Module Name stock_count_purge_job

Description Purge Stock Count

Functional Area Stock Counts

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (STOCK_COUNT_PURGE_THREAD) will filter eligible records from stake count head (STAKE_HEAD) table based on its purge criteria from system variable settings. The Last End-of-Month Start Month (last_eom_start_month) parameter will determine records with earlier stock take date or those that have been flagged for deletion. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_STOCK_COUNT_PURGE_STG.

The Business logic program (STOCK_COUNT_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from stock count related tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 17–8 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day

Table 17–8 (Cont.) Scheduling Constraints

Schedule Information	Description
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Cycle Count

Restart/Recovery

NA

Key Tables Affected

Table 17–9 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_VARIABLES	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_STOCK_COUNT_ PURGE_STG	Yes	Yes	No	Yes
STAKE_LOCATION	Yes	No	No	Yes
STAKE_QTY	No	No	No	Yes
STAKE_CONT	No	No	No	Yes
STAKE_SKU_LOC	No	No	No	Yes
STAKE_PROD_LOC	No	No	No	Yes
STAKE_PRODUCT	No	No	No	Yes
STAKE_HEAD	Yes	No	No	Yes

Design Assumption

NA

stkschedxpld (Create Stock Count Requests Based on Schedules)

Module Name	stkschedxpld.pc
Description	Create Stock Count Requests Based on Schedules
Functional Area	Stock Counts
Module Type	Business Processing
Module Technology	ProC
Catalog ID	N/A
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch process is used to create stock count requests based on pre-defined schedules for a location. It evaluates all scheduled counts, that are planned for x number of days from the current day. The number of days prior to the planned count date by which the count requests are created is determined by the system parameter Stock Count Review Days.

For Unit counts, the item list specified is exploded out to the transaction-level and written to the count/item/location table. For Unit & Value counts, the transaction-level items contained in the specified department/class/subclass will be written to the count/item/location table and count/product/location tables. If the schedule was created using a location list, then this process also explodes that down to the store or virtual warehouse level.

Restart/Recovery

The logical unit of work for this module is schedule, location. The changes will be posted when the maximum commit counter value is reached.

Key Tables Affected

Table 17–10 Key Tables Affected

Table	Select	Insert	Update	Delete
STAKE_SCHEDULE	Yes	No	Yes	No
V_RESTART_STORE_WH	Yes	No	No	No
PERIOD	Yes	No	No	No
CODE_DETAIL	Yes	No	No	No
STAKE_HEAD	No	Yes	No	No
STAKE_LOCATION	No	Yes	No	No
STAKE_PRODUCT	No	Yes	No	No
STAKE_PROD_LOC	No	Yes	No	No
STAKE_SKU_LOC	Yes	Yes	No	No
ITEM_MASTER	Yes	No	No	No
DEPS	Yes	No	No	No
SUBCLASS	Yes	No	No	No
PACKITEM	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
SKULIST_DETAIL	Yes	No	No	No
LOC_LIST_DETAIL	Yes	No	No	No
LOCATION_CLOSED	Yes	No	No	No
COMPANY_CLOSED	Yes	No	No	No

Table 17–10 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
INV_TRACK_UNIT_OPTIONS	Yes	No	No	No

Design Assumption

N/A

stake_sched_explode_job (Create Stock Count Requests Based on Schedules)

Module Name	stake_sched_explode_job
Description	Create Stock Count Requests Based on Schedules
Functional Area	Stock Counts
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (STKSCHED_EXPLODE_THREAD) will filter eligible records from stake count schedule (STAKE_SCHEDULE) and store or location list (STORE, LOC_LIST_HEAD, LOC_LIST_DETAIL) tables based on its review criteria from system parameter settings. The Stake Count Review Days (stake_review_days) parameter will determine and evaluate scheduled counts that are planned for x days from the current day. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_STKSCHED_EXPLODE_STG.

The Business logic program (STKSCHED_EXPLODE) will process all records from the staging table. Using bulk processing, this program will create stock count requests based in the pre-defined schedules for a location. For Unit counts, the item list specified is exploded out to the transaction-level and written to the count/item/location (STAKE_SKU_LOC) table. For Unit & Value counts, the transaction-level items contained in the specified department/class/subclass will be written to the count/item/location (STAKE_SKU_LOC) and count/product/location (STAKE_PROD_LOC) tables. If the schedule was created using a location list, then this process also explodes that down to the store or virtual warehouse level. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 17–11 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Schedule ID

Restart/Recovery

NA

Key Tables Affected

Table 17–12 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_STKSCHED_ EXPLODE_STG	Yes	Yes	No	Yes
STAKE_SCHEDULE	Yes	No	Yes	No
V_RESTART_STORE_WH	Yes	No	No	No
PERIOD	Yes	No	No	No
CODE_DETAIL	Yes	No	No	No
STAKE_HEAD	No	Yes	No	No
STAKE_LOCATION	No	Yes	No	No
STAKE_PRODUCT	No	Yes	No	No
STAKE_PROD_LOC	No	Yes	No	No
STAKE_SKU_LOC	Yes	Yes	No	No
ITEM_MASTER	Yes	No	No	No
DEPS	Yes	No	No	No
SUBCLASS	Yes	No	No	No
PACKITEM	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
SKULIST_DETAIL	Yes	No	No	No
LOC_LIST_DETAIL	Yes	No	No	No
LOCATION_CLOSED	Yes	No	No	No
COMPANY_CLOSED	Yes	No	No	No

Table 17–12 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
INV_TRACK_UNIT_ OPTIONS	Yes	No	No	No

Design Assumption

NA

stkupd (Stock Count Snapshot Update)

Module Name	stkupd.pc
Description	Stock Count Snapshot Update
Functional Area	Stock Counts
Module Type	Business Processing
Module Technology	ProC
Integration Catalog ID	RMS362
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Stock Count Snapshot Update is a nightly batch program used to take a 'snapshot' of inventory, cost and retail values prior to the count commencing. This will be used to calculate the book value of the count. The stock count snapshot includes stock on hand, in-transit-qty, cost (either WAC or standard cost, based on system settings) and retail for each item-location record. The snapshot is taken on the day that the count is scheduled. Additionally, transaction data snapshots of future-dated transactions are captured and stored in a table that will be used by Stock Count Shrinkage Update batch.

Restart/Recovery

This program is multithread using the restart all locations view. The logical unit of work is an item/location.

Key Tables Affected

Table 17–13 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
STAKE_SKU_LOC	Yes	No	Yes	No

Table 17–13 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
STAKE_HEAD	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
TRAN_DATA_FUTURE_SNAPSHOT	No	Yes	No	No
TRAN_DATA	Yes	No	No	No

Design Assumption

N/A

stkvar (Update Stock On Hand Based on Stock Count Results)

Module Name	stkvar.pc
Description	Update Stock On Hand Based on Stock Count Results
Functional Area	Stock Counts
Module Type	Business Processing
Module Technology	ProC
Integration Catalog ID	RMS363
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Stock Count Stock on Hand Updates batch process updates stock on hand based on the unit count results. For Unit counts, it also writes transaction data records for any variances to transaction code 22. For Unit & Value counts, it also computes the total cost and total retail value of the count and updates the count/product/location table with this information. The post processing for this batch inserts dept/class/subclass/location records into the week, month and half data tables in cases wherein they don't exist.

Restart/Recovery

The logical unit of work for this program is item, location type and location. This program is multithread using the restart stock count view. After the maximum commit counter number of rows is processed, intermittent commits are done to the database and the item/location information is written to restart tables for restart/recovery.

Key Tables Affected

Table 17–14 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
PERIOD	Yes	No	No	No
ITEM_XFORM_HEAD	Yes	No	No	No
ITEM_XFORM_DETAIL	Yes	No	No	No
STAKE_SKU_LOC	Yes	No	Yes	No
STAKE_CONT	Yes	No	No	Yes
STAKE_HEAD	Yes	No	No	No
STAKE_CONT_TEMP	Yes	Yes	No	Yes
STAKE_PROD_LOC	Yes	No	Yes	No
WH	Yes	No	No	No
CLASS	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	Yes	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
EDI_DAILY_SALES	No	No	Yes	No
TRAN_DATA	No	Yes	No	No
NWP	No	Yes	Yes	No
NWP_FREEZE_DATE	Yes	No	No	No
STAKE_QTY	Yes	No	No	No
STAKE_LOCATION	Yes	No	No	No
STAKE_PRODUCT	Yes	No	No	No
STORE	Yes	No	No	No
VAT_ITEM	Yes	No	No	No

Design Assumption

N/A

stkxpld (Explode Stock Count Requests to Item Level)

Module Name stkxpld.pc

Description Explode Stock Count Requests to Item Level

Functional Area Stock Counts

Module Type Business Processing

Module Technology ProC

Catalog ID RMS364

Wrapper Script rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Stock Count Explode batch is a nightly batch is used to explode stock count requests created at the department, class or subclass level to the item level. This process must run before the stock count snapshot is taken and is run for counts x days prior to the count based on the system parameter setting, Stock Count Lockout Days.

The batch process picks up product groups (departments, classes or subclasses) from the count/product table and inserts records into the count/item/location table and the count/product/location table (for Unit & Value counts) for all items in the product group that exist for the locations on the count. Only approved inventoried items are added to stock counts.

For transformable items, both the non-inventoried sellable items and inventoried orderable items that are contained in a product group will also be added to the count. For deposit items, only the content, crate and packs can be counted.

Restart/Recovery

This batch program is multithreaded using the restart all locations view. The logical unit of work for this program is a cycle count/location.

Key Tables Affected

Table 17–15 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
STAKE_LOCATION	Yes	No	No	No
STAKE_HEAD	Yes	No	No	No
STAKE_SKU_LOC	Yes	Yes	No	No
STAKE_PROD_LOC	Yes	Yes	No	No
STAKE_PRODUCT	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
ITEM_XFORM_HEAD	Yes	No	No	No
ITEM_XFORM_DETAIL	Yes	No	No	No
SUBCLASS	Yes	No	No	No

Design Assumption

N/A

stockcountprocess.ksh (Process Stock Count Results)

Module Name	stockcountprocess.ksh
Description	Process Stock Count Results
Functional Area	Stock Counts
Module Type	Business Processing
Module Technology	ksh
Integration Catalog ID	RMS366
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Stock Count Process batch processes actual count data from the selected store or physical warehouse to count/item/location table from the data staged by STOCKCOUNTUPLOAD.KSH. For a physical warehouse, this process also calls the Merchandising distribution library to apportion quantities to the virtual warehouses in Merchandising.

Restart/Recovery

The logical unit of work for stockcountprocess.ksh is a set of a single or multiple valid items at a given location. This set is defined as a chunk. Based on the example above, if for some reason, chunk 2 raised an error, INPUT FILE 6, 7, and 8 wouldn't be processed by this program. Other chunks, if there are no errors, would be processed. User has to correct the transaction details and upload the input file again that includes the affected CHUNKS for reprocessing.

Key Tables Affected

Table 17–16 Key Tables Affected

Table	Select	Insert	Update	Delete
STK_FILE_STG	Yes	Yes	No	No
STAKE_SKU_LOC	Yes	Yes	Yes	No
STK_SSL_TEMP	Yes	Yes	No	No
STAKE_QTY	Yes	Yes	Yes	Yes
WH	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
STK_SSL_TEMP	Yes	Yes	No	No
STK_XFORM_TEMP	Yes	Yes	No	No
STAKE_PROD_LOC	Yes	No	No	No

Table 17–16 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
STAKE_PRODUCT	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
STAKE_PROD_LOC	Yes	No	No	No
ITEM_XFORM_DETAIL	Yes	No	No	No
ITEM_XFORM_HEAD	Yes	No	No	No
STK_XFORM_ORD_TEMP	Yes	Yes	No	No
STAKE_LOCATION	Yes	Yes	No	No
PARTNER	Yes	No	No	No
STAKE_HEAD	Yes	No	No	No
STK_DUP_SQT_TEMP	Yes	Yes	No	No
WORK_STKUPLD_STAKE_QTY_GTT	Yes	Yes	Yes	Yes
WORK_STKUPLD_ITEM_LOC_GTT	Yes	Yes	Yes	Yes
FILE_UPLOAD_STATUS	Yes	No	Yes	No
FILE_UPLOAD_ERRORS	Yes	Yes	Yes	No

Design Assumption

N/A

18

Oracle Retail Trade Management

Trade Management automates international import transaction data. There are six components of Trade Management:

- Customs entry
- Harmonized tariff schedule
- Letter of credit
- Transportation
- Actual landed costs
- Obligations

Four of these components—customs entry, Harmonized Tariff Schedule, letter of credit, and transportation—have batch-processing modules that facilitate the flow of data between Trade Management and external applications and files. This chapter describes these batch modules, along with Perl scripts, and the kinds of data that they process.

For additional information about Trade Management, including detailed flow diagrams, see the Oracle Retail Merchandising Functional Library (Doc ID: 1585843.1).

> **Note:** The White Papers in this library are intended only for reference and educational purposes and may not reflect the latest version of Oracle Retail software.

Simplified Trade Management Configuration

Simplified Trade Management is a simplified version of the Oracle Retail product suite targeted at mid-tier retailers. The Simplified Oracle Retail Merchandising Operations Management applications support basic retail processes needed by a mid-tier retailer. Advanced features are turned-off through system parameters, with the goal to reduce implementation complexity and enabling faster implementation and lower total cost of ownership.

The Simplified Trade Management Indicator is a system parameter set during the installation of Merchandising. If the system parameter is enabled, then the following Trade Management functionality is not available in the application:

- Setting up Trade Management specific Freight Type, Freight Size and Standard Carrier Alpha Codes (SCAC)
- Letter of Credit functionality
- Transportation functionality

- Customs Entry functionality
- Obligation Maintenance
- Actual Landed Costs

If both the Simplified Trade Management indicator and the Import indicator are enabled, then some import related functionality is available in Merchandising. With this setup, the retailer has the option to setup HTS data for classification of merchandise and for the calculation of duties, fee and taxes.

The retailer can also choose Letter of Credit as a payment option at the Purchase Order header level, but all other related LC functionality is not available. It is assumed that the retailer is using some other external system for LC processing.

If the import indicator is not enabled then no Trade Management functionality is available in the application. See the Merchandising Installation Guide for additional information on setting the value of the system parameter.

Simplified Trade Management Batch Program Notes

When Simplified Trade Management is enabled (Trade Management Simplified Indicator is enabled) then the following batch programs need to be turned off from the integrated batch schedule.

- lcadnld
- lcupld
- lcup798
- lcmdnld
- cednld
- tranupld

The following Perl scripts should also be turned off from the integrated batch schedule.

- lcmt700
- lcmt707
- lcmt730
- lcmt798

When both the Trade Management simplified indicator and import indicator is enabled then the following batch program needs to be turned on in the integrated batch schedule.

- htsupld

Batch Design Summary

The following batch designs are included in this functional area:

- cednld.pc (Download of Customs Entry Transactions to Brokers)
- htsupld.pc (Harmonized Tariff Schedule Upload)
- tranupld.pc (Transportation Upload)
- lcadnld.pc (Letter of Credit Application Download)

- lcmt700 Perl (SWIFT File Conversion – Letter of Credit Application)
- lcupld.pc (Letter of Credit Confirmation Upload)
- lcmt730 (SWIFT File Conversion – Letter of Credit Confirmation)
- lcmdnld.pc (Letter of Credit Amendment Download)
- lcmt707 Perl (SWIFT File Conversion – Letter of Credit Amendment)
- lcup798.pc (Letter of Credit Drawdowns and Charges)
- lcmt798 (SWIFT File Conversion – Letter of Credit Charges and Drawdowns)

cednld (Download of Customs Entry Transactions to Brokers)

Module Name	cednld.pc
Description	Download of Customs Entry Transactions to Brokers
Functional Area	Oracle Retail Trade Management
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS53
Wrapper Script	batch_cednld.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is used to download custom entry information from the Merchandising database to brokers. Each night, this program reads all customs entry (CE) transactions that are in Sent status for a broker ID. These transactions are written to a flat file and the status is changed to Downloaded. One flat file is written per broker.

Restart/Recovery

The Logical Unit of Work for the program is a single row from the customs entry header table. Restart/Recovery will be used for init and commit.

Table based restart/recovery must be used. The commit max counter field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O. The recommended commit counter setting is 1000 records (subject to change based on implementation).

Key Tables Affected

Table 18–1 Key Tables Affected

Table	Select	Insert	Update	Delete
CE_HEAD	Yes	No	Yes	No
CE_SHIPMENT	Yes	No	No	No
CE_ORD_ITEM	Yes	No	No	No

Table 18–1 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ORDHEAD	Yes	No	No	No
SUP_IMPORT_ATTR	Yes	No	No	No
TRANSPORTATION	Yes	No	No	No
CE_LIC_VISA	Yes	No	No	No
CE_CHARGES	Yes	No	No	No
MISSING_DOC	Yes	No	No	No

I/O Specification

Integration Type Download from Merchandising

File Name Determined by runtime parameter

Integratin Contract IntCon000050

Output File Layout

Table 18–2 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Identifier	Number(10)	Nine leading zeroes: 0000000001	ID of current line being processed by input file
	File Type Definition	Char(4)	CEDN	Identifies file as 'Customs Entry download'
	File Create Date	Date	Create date	Vdate in YYYYMMDD HH24MISS format
THEAD	File Type Descriptor	Char(5)	THEAD	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	CE ID	Number(10)	ce_head.ce_id	N/A
	Entry No	Char (15)	ce_head.entry_no	N/A
	Entry Date	Char(14)	ce_head.entry_date	YYYYMMDDHH24MISS format
	Entry Status	Char(6)	ce_head.entry_status	N/A
	Entry Type	Char(6)	ce_head.entry_type	N/A
	Entry Port	Char(5)	ce_head.entry_port	N/A
	Summary Date	Char(14)	ce_head.summary date	YYYYMMDDHH24MISS format
	Broker ID	Char(10)	ce_head.broker_id	N/A
	Broker Ref. ID	Char(18)	ce_head.broker_ref_id	N/A
	File Number	Char(18)	ce_head.file_no	N/A
	Importer ID	Char(10)	ce_head.importer_id	N/A
	Import Country	Char(3)	ce_head.import_country_id	N/A
	Currency Code	Char(3)	ce_head.currency_code	N/A
	Exchange Rate	Number(20,10)	ce_head.exchange_rate*10000000000 (with 10 implied decimal places)	N/A
	Bond Number	Char(18)	ce_head.bond_no	N/A
	Bond Type	Char(6)	ce_head.bond_type	N/A

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Surety Code	Char(6)	ce_head.surety_code	N/A
	Consignee ID	Char(10)	ce_head.consignee_id	N/A
	Live Indicator	Char(1)	ce_head.live_ind	N/A
	Batch Number	Char(20)	ce_head.batch_no	N/A
	Entry Team	Char(3)	ce_head.entry_team	N/A
	Liquidation Amount	Number(20,4)	ce_head.liquidation_amt*10000 (4 implied decimal places)	N/A
	Liquidation Date	Date	ce_head.liquidation_date	YYYYMMDDHH24MISS format
	Reliquidation Amount	Number(20,4)	ce_head.reliquidation_amt*10000 (4 implied decimal places)	N/A
	Reliquidation Date	Date	ce_head.reliquidation_date	YYYYMMDDHH24MISS format
	Merchandise Loc	Char(40)	ce_head.merchandise_loc	N/A
	Location Code	Char(4)	ce_head.location_code	N/A

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TSHIP	File Type Descriptor	Char(5)	TSHIP	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file
	Vessel ID	Char(20)	ce_shipment.vessel_id	N/A
	Voyage Flt ID	Char(10)	ce_shipment.voyage_flt_id	N/A
	Estimated Departure Date	Date	ce_shipment.estimated_depart_date	YYYYMMDDHH24MISS format
	Vessel SCAC Code	Char(6)	ce_shipment.vessel_scac_code	N/A
	Lading Port	Char(5)	ce_shipment.lading_port	N/A
	Discharge Port	Char(5)	ce_shipment.discharge_port	N/A
	Tran Mode ID	Char(6)	ce_shipment.tran_mode_id	N/A
	Export Date	Date	ce_shipment.export_date	YYYYMMDDHH24MISS
	Import Date	Date	ce_shipment.import_date	YYYYMMDDHH24MISS
	Arrival Date	Date	ce_shipment.arrival_date	YYYYMMDDHH24MISS
	Export Country	Char(3)	ce_shipment.export_country_id	N/A
	Shipment Number	Number(10)	ce_shipment.shipment_no	N/A
TORDI	File Type Descriptor	Char(5)	TORDI	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Order Number	Number(8)	ce_ord_item.order_no	N/A
	Item	Char (25)	ce_ord_item.item	N/A
	BL AWB ID	Char(30)	ce_ord_item.bl_awb_id	'MULTI' – means multiple airway bills (otherwise a single airway bill will be retrieved)
	Invoice ID	Char(30)	ce_ord_item.invoice_id	N/A
	Invoice Date	Date	ce_ord_item.invoice_date	YYYYMMDDHH24MISS format
	Invoice Amount	Number(20,4)	ce_ord_item.invoice_amt*10000 (4 implied decimal places)	N/A
	Currency Code	Char(3)	ce_ord_item.currency_code	N/A
	Exchange Rate	Number(20,10)	ce_ord_item.exchange_rate*10000000000 (10 implied decimal places)	N/A
	Manifest Item Quantity	Number(12,4)	ce_ord_item.manifest_item_qty*10000 (4 implied decimal places)	N/A
	Manifest Item Quantity UOM	Char(4)	ce_ord_item.manifest_item_qty_uom	N/A
	Carton Quantity	Number (12,4)	ce_ord_item.carton_qty*10000 (4 implied decimal places)	N/A

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Carton Quantity UOM	Char(4)	ce_ord_item.carton_qty_uom	N/A
	Gross Weight	Number(12,4)	ce_ord_item.gross_wt*10000 (4 implied decimal places)	N/A
	Gross Weight UOM	Char(4)	ce_ord_item.gross_wt_uom	N/A
	Net Weight	Number(12,4)	ce_ord_item.net_wt*10000 (4 implied decimal places)	N/A
	Net Weight UOM	Char(4)	ce_ord_item.net_wt_uom	N/A
	Cubic	Number(12,4)	ce_ord_item.cubic*10000 (4 implied decimal places)	N/A
	Cubic UOM	Char(4)	ce_ord_item.cubic_uom	N/A
	Cleared Quantity	Number(12,4)	ce_ord_item.cleared_qty*10000 (4 implied decimal places)	N/A
	Cleared Quantity UOM	Char(4)	ce_ord_item.cleared_qty_uom	N/A
	In Transit Number	Char(15)	ce_ord_item.in_transit_no	N/A
	In Transit Date	Date	ce_ord_item.in_transit_date	YYYYMMDDHH24MISS format
	Rush Indicator	Char(1)	ce_ord_item.rush_ind	N/A
	Related Indicator	Char(1)	ce_ord_item.related_ind	N/A

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Tariff Treatment	Char(10)	ce_ord_item.tariff_treatment	N/A
	Ruling Number	Char(10)	ce_ord_item.ruling_no	N/A
	Do Number	Char(10)	ce_ord_item.do_no	N/A
	Do Date	Date	ce_ord_item.do_date	YYYYMMDDHH24MISS format
	Manufacture ID	Char(18)	sup_import_attr.mfg_id	N/A
TBLAW	File Type Descriptor	Char(5)	TBLAW	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file
	BL AWB ID	Char(30)	Transportation.bl_awb_id	N/A
TCONT	File Type Descriptor	Char(5)	TCONT	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file
	Container ID	Char(20)	Transportation.container_id	N/A
	Container SCAC Code	Char(6)	Transportation.container_scac_code	N/A

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TLICV	File Type Descriptor	Char(5)	TLICV	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file
	License/Visa Type	Char(6)	ce_lic_visa.license_visa_type	N/A
	License/Visa ID	Char(30)	ce_lic_visa.license_visa_id	N/A
	License/Visa Quantity	Number(12,4)	ce_lic_visa.license_visa_qty*10000 (4 implied decimal places)	N/A
	License/Visa Quantity UOM	Char(4)	ce_lic_visa.license_visa_qty_uom	N/A
	Quota Category	Char (6)	ce_lic_visa.quota_category	N/A
	Net Weight	Number(12,4)	ce_lic_visa.net_weight*10000 (4 implied decimal places)	N/A
	Net Weight UOM	Char(4)	ce_lic_visa.net_weight_uom	N/A
	Holder ID	Char(18)	ce_lic_visa.holder_id	N/A

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TCHRG	File Type Descriptor	Char(5)	TCHRG	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file
	Sequence Number	Number(6)	ce_ charges.seq_ no	N/A
	Pack Item	Char(25)	ce_ charges.pack _item	N/A
	HTS	Char(10)	ce_ charges.hts	N/A
	Effect From Date	Date	ce_ charges.effect _from	YYYYMMDD HH24MISS format
	Effect To Date	Char(14)	ce_ charges.effect _to	YYYYMMDD HH24MISS format
	Component ID	Date	ce_ charges.com p_id	N/A
	Component Rate	Number(20,4)	ce_ charges.com p_rate*10000 (4 implied decimal places)	N/A
	Per Count UOM	Char(3)	ce_ charges.per_ count_uom	N/A
	Component Value	Number(20,4)	ce_ charges.com p_value * 10000 (4 implied decimal places)	N/A
TMDOC	File Type Descriptor	Char(5)	TMDOC	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file
	Doc_id	Number(6)	Missing_ doc.doc_id	N/A
	Received_ date	Date	Missing_ doc.received _date	YYYYMMDD HH24MISS format

Table 18–2 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FTAIL	File Type Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Identifier	Number(10)	Incremented internally	ID of current line being processed by input file.
	File Record Counter	Number(10)	Determined internally	Number of records/transactions processed in current file (only records between head & tail)

Design Assumptions

N/A

htsupld (Harmonized Tariff Schedule Upload)

Module Name	htsupld.pc
Description	Harmonized Tariff Schedule Upload
Functional Area	Oracle Retail Trade Management
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS41
Wrapper Script	rmswrap_multi_in_rej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The harmonized tariff schedule module processes a file containing the most recent United States Customs tariff schedule to Merchandising tables. The module uploads both the initial entry of the schedule and all the updates, as they become available.

Restart/Recovery

Recommended commit counter is 2000. Input file names must end in a ".1" for the restart mechanism to properly parse the file name. Because there is only 1 input file to be uploaded, only 1 thread is used.

A reject file is used to hold records that have failed processing. You can fix the rejected records and process the reject file again.

Key Tables Affected

Table 18–3 Key Tables Affected

Table	Select	Insert	Update	Delete
HTS	Yes	Yes	Yes	Yes
HTS_TL	No	No	No	Yes
HTS_TARIFF_TREATMENT	Yes	Yes	Yes	Yes
ITEM_HTS	Yes	Yes	Yes	Yes
MOD_ORDER_ITEM_HTS	No	Yes	No	No
HTS_OGA	No	Yes	Yes	Yes
ORDSKU_HTS	Yes	Yes	Yes	Yes
HTS_TT_EXCLUSIONS	No	Yes	Yes	Yes
HTS_TAX	No	Yes	Yes	Yes
HTS_FEE	No	Yes	Yes	Yes
CE_CHARGES	Yes	Yes	Yes	Yes
HTS_CHAPTER	Yes	Yes	No	No
QUOTA_CATEGORY	Yes	Yes	No	No
ITEM_HTS_ASSESS	No	Yes	Yes	Yes
HTS_AD	No	No	Yes	No
HTS_CVD	No	No	Yes	No
HTS_REFERENCE	No	No	Yes	No
ORDHEAD	Yes	No	Yes	No
ITEM_EXP_DETAIL	No	No	Yes	No
ORDLOC_EXP	No	No	Yes	No
ORDSKU_HTS_ASSESS	No	No	Yes	Yes
ORDSKU_TEMP	Yes	No	No	Yes
ORDLOC_TEMP	No	No	No	Yes
ALLOC_CHRG_TEMP	No	No	No	Yes
ALLOC_DETAIL_TEMP	No	No	No	Yes
ALLOC_HEADER_TEMP	No	No	No	Yes
ORDLOC_EXP_TEMP	No	No	No	Yes
ORDSKU_HTS_ASSESS_ TEMP	No	No	No	Yes
ORDSKU_HTS_TEMP	No	No	No	Yes
ORDLOC_DISCOUNT_ TEMP	No	No	No	Yes
TIMELINE_TEMP	No	No	No	Yes
REQ_DOC_TEMP	No	No	No	Yes
WO_DETAIL_TEMP	No	No	No	Yes
WO_HEAD_TEMP	No	No	No	Yes

Table 18–3 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
REPL_RESULTS_TEMP	No	No	No	Yes

I/O Specification

Integration Type Upload to Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000051

Input File Layout

Table 18–4 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record Descriptor	Char(5)	FHEAD	Describes file line type
	Line number	Number(10)	0000000001	Sequential file line number
	File ID	Char(5)	HTSUP	Describes file type
THEAD	Record Descriptor	Char(5)	THEAD	Describes file line type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction id	Number(14)	N/A	Unique transaction id
	HTS Line	Char(358)	N/A	V1 through V4 records from the customs HTS file concatenated together

Table 18–4 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
TDETL	Record Descriptor	Char(5)	TDETL	Describes file line type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction id	Number(10)	N/A	Unique transaction id
	Tax/fee line	Char(80)	N/A	V5 through VC records from the customs HTS file, each on a separate TDETL line
TTAIL	Record Descriptor	Char(5)	TTAIL	Describes file line type
	Line number	Number(10)	N/A	Sequential file line number
	Detail lines	Number(6)	N/A	Number of lines between THEAD and TTAIL
FTAIL	Record Descriptor	Char(5)	FTAIL	Describes file line type
	Line number	Number(10)	N/A	Sequential file line number
	Transaction Lines	Number(10)	N/A	Number of lines between FHEAD and FTAIL

Original Input File

Note: The input file contains lines of 2400 characters (that is, the newline character occurs only after every 2400 characters). Each 2400-character line consists of thirty 80-character records. Each 80-character record starts with 'V1' or 'V2' … or 'VD' or blank if the record is completely empty. For each tariff, records V1 and V2 are mandatory; records V3 through VD are optional, which means they can be all blank. Record V4 is not currently used in Merchandising/Trade Management. Records V5 through VC contain the tax/fee information for the tariff, and all have the same structure. The lower-case letters in the record name block are as a convenience to cross-reference with the US Customs file description.

Table 18–5 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
V1 a	Control identifier	Char(1)	V	Identifies start of record
b	Record type	Char(1)	1	Identifies record type
c	Tariff number	Number(10)	N/A	A code located in the Harmonized Tariff Schedule of the United States Annotated (HTS) representing the tariff number. If this number is less than 10 positions, it is left justified
d	Transaction code	Char(1)	A,D,R	A code representing the type of transaction. Valid Transaction Codes are: A = Add D = Delete R = Replace
e	Beginn effective date	char(6)	N/A	A numeric date in MMDDYY (month, day, year) format representing the record begin effective date. This date indicates when the record becomes effective
f	End effective date	char(6)	N/A	A numeric date in MMDDYY (month, day, year) format representing the record end effective date. This date indicates the last date the record is effective

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
g	number of reporting units	number(1)	0,1, or 2 or 3	The number of reporting units required by the Bureau of the Census. In a few instances, units not required by Census may be required to compute duty. In these cases, the Census reporting units are always first, followed by any additional units required to compute the duty
h	1st reporting unit of measure	char(4)	N/A	A code representing the first unit of measure. If the reporting unit is X, no unit of measure is required except for certain tariff numbers in Chapter 99. Valid unit of measure codes are listed in Appendix C
I	2nd reporting unit of measure	char(4)	N/A	A code representing the second unit of measure. Valid unit of measure codes are listed in Appendix C
j	3rd reporting unit of measure	char(4)	N/A	A code representing the third unit of measure. Valid unit of measure codes are listed in Appendix C

Table 18–5 (Cont.) Input File Layout

Record Name		Field Name	Field Type	Default Value	Description
k		duty computation code	char(1)	N/A	A code indicating the formula to be used to compute the duty. Valid Duty Computation Codes are listed in Appendix F
l		commodity description	char(30)	N/A	A condensed version of the commodity description that appears in the HTS
m		column 1 specific rate of duty	Number(12)	N/A	The rate of duty that appears in the General column of the HTS. Eight decimal places are implied
n		base rate indicator	char(1)	'B' or blank	A code indicating if the rate contains a base rate. If the base rate indicator is B, the duty rate is a base rate; otherwise, space fill. Not Used in Merchandising
o		space fill	char(1)	blank	Space fill. Not used in Merchandising
V2	a	Control identifier	char(1)	V	Identifies start of record
	b	Record type	char(1)	2	Identifies record type

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
c	tariff number	Number (10)	N/A	A code located in the Harmonized Tariff Schedule of the United States Annotated (HTS) representing the tariff number. If this number is less than 10 positions, it is left justified. This number is the same as that in Record Identifier V1
d	general column 1 ad valorem percentage	Number (12)	N/A	The ad valorem rate of duty that appears in the General column of the HTS. Eight decimal places are implied
e	column 1 other	Number (12)	N/A	The rate of duty that appears in the General column of the HTS that is not an ad valorem rate. Eight decimal places are implied
f	Column 2 specific rate	Num(12)	N/A	The specific rate of duty that appears in Column 2 of the HTS. Eight decimal places are implied
g	Column 2 ad valorem percentage	Num(12)	N/A	The ad valorem rate of duty that appears in Column 2 of the HTS. Eight decimal places are implied

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
h	Column 2 other rate	Num(12)	N/A	The rate of duty that appears in Column 2 of the HTS that is not an ad valorem rate or a specific rate. Eight decimal places are implied
i	countervailing duty flag	char(1)	blank or 1	A code of 1 indicating the tariff number is subject to countervailing duty; otherwise, space fill
j	additional tariff indicator	char(1)	blank or 'R'	A code indicating if an additional tariff number may be required with this tariff number. Refer to the Harmonized Tariff Schedule of the United States Annotated (HTS) for more specific information on which HTS numbers require additional HTS numbers to be reported. This indicator is R when an additional tariff number may be required; otherwise, space fill
k	Miscellaneous Permit/ License Indicator	char(2)	N/A	A code indicating if a tariff number may be subject to a miscellaneous permit/license number

Table 18–5 (Cont.) Input File Layout

Record Name		Field Name	Field Type	Default Value	Description
l		space fill	char(4)	blanks	Not used in Merchandising
V3	a	Control identifier	char(1)	V	identifies start of record
	b	Record type	char(1)	3	identifies record type
	c	tariff number	Number(10)	N/A	A code located in the Harmonized Tariff Schedule of the United States Annotated (HTS) representing the tariff number. If this number is less than 10 positions, it is left justified. This number is the same as the number in Record Identifier V1
	d	GSP excluded countries	char(20)	N/A	The International Organization for Standardizatio n (ISO) country code that indicates countries not eligible for preferential treatment under GSP. Up to ten 2 position country codes can be reported. If countries are excluded from GSP, the Special Programs Indicator (SPI) Code contained in this record (positions 53 64) is A*. Valid ISO country codes are listed in Appendix B

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
e	OGA codes	char(15)	N/A	Codes that indicate special requirements by other Federal Government agencies must or may apply. Up to five 3 position OGA codes can be provided
f	anti-dumping flag	char(1)	1 or blank	A code of 1 indicating the tariff number is subject to an antidumping duty; otherwise, space fill
g	quota indicator	char(1)	1 or blank	A code of 1 indicating the tariff number may be subject to quota. If the tariff number is not subject to quota, space fill
h	category number	char(6)	N/A	A code located in the HTS indicating the textile category assigned to the tariff number. If there is no textile category number, space fill

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
I	special program indicators	char(28)	N/A	A code indicating if a tariff number is subject to a special program. Up to fourteen 2 position codes can be reported. Left justify. The SPI codes are not reported in any particular sequence. If more than fourteen 2-position codes are required, they are reported on the VD record
NEWLINE			\n	
V4 a	Control identifier	char(1)	V	identifies start of record. Entire V4 record not used in Merchandising
b	Record type	char(1)	4	identifies record type
c	tariff number	Number (10)	N/A	A code located in the Harmonized Tariff Schedule of the United States Annotated (HTS) representing the tariff number. If this number is less than 10 positions, it is left justified. This number is the same as the number reported in Record Identifier V1
d	value edit code	char(3)	N/A	A code representing the value edit

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
e	value low bounds	Number (10)	N/A	A value representing the minimum value edit. Five decimal places are implied. If this record contains date edits (positions 36 53), space fill
f	value high bounds	Number (10)	N/A	A value representing the maximum value edit. Five decimal places are implied. If this record contains date edits (positions 36 53), space fill
g	entry date restriction	Number (1)	0,1, or 2	A code representing the first entry date restriction code
h	beginning restriction date	char(4)	N/A	A numeric date in MMDD (month and day) format representing the first begin restriction date used in the edit. If this record contains a value edit (positions 13 35), space fil
l	end restriction date	char(4)	N/A	A numeric date in MMDD (month and day) format representing the first end restriction date used in the edit. If this record contains a value edit (positions 13 35), space fill

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
j	entry date restriction 2	number(1)	0,1 or 2	A code representing the second entry date restriction code
k	beginning restriction date 2	char(4)	N/A	A numeric date in MMDD (month and day) format representing the second begin restriction date used in the edit. If this record contains a value edit (positions 13 35), space fill
l	end restriction date 2	char(4)	N/A	A code located in the Harmonized Tariff Schedule of the United States Annotated (HTS) representing the tariff number. If this number is less than 10 positions, it is left justified. This number is the same as the number reported in Record Identifier V1
m	country of origin	char(2)	N/A	A code representing the value edit
n	space filler	char(2)	blanks	A value representing the minimum value edit. Five decimal places are implied. If this record contains date edits (positions 36 53), space fill

Table 18–5 (Cont.) Input File Layout

Record Name		Field Name	Field Type	Default Value	Description
o		quantity edit code	char(3)	N/A	A value representing the maximum value edit. Five decimal places are implied. If this record contains date edits (positions 36 53), space fill
p		low quantity	Number (10)	N/A	A code representing the first entry date restriction code
q		high quantity	Number (10)	N/A	A numeric date in MMDD (month and day) format representing the first begin restriction date used in the edit. If this record contains a value edit (positions 13 35), space fill
V5	a	Control identifier	char(1)	V	Identifies start of record
b		Record type	char(1)	5,6,7,8,9,A,B, C	Identifies record type
c		tariff number	Number (10)	N/A	A code located in the Harmonized Tariff Schedule of the United States Annotated (HTS) representing the tariff number. If this number contains less than 10 positions, it is left justified. This number is the same as the number reported in Record Identifier V1

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
d	Country code	char(2)	N/A	A code representing the country. Valid ISO country codes are listed in Appendix B. E followed by a space (Caribbean Basin Initiative), and J followed by a space (Andian Trade Preference Act), and R followed by a space (Caribbean Trade Partnership Act), are also valid codes for special rates. Countries eligible for E and J are indicated in the ACS country code file and the Harmonized Tariff Schedule of the United States Annotated (HTS)
e	specific rate	Number (12)	N/A	The specific rate of duty listed in the Special column of the HTS. Eight decimal places are implied
f	ad valorem rate	Number (12)	N/A	The ad valorem rate of duty listed in the Special column of the HTS. Eight decimal places are implied

Table 18–5 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
g	Other rate	Number (12)	N/A	The rate of duty listed in the Special column of the HTS that is not a specific or ad valorem rate. Eight decimal places are implied
h	tax/fee class code	char(3)	N/A	A code representing the tax/fee class. Valid tax/fee class codes are listed in Appendix B
I	tax/fee comp code	char(1)	N/A	A code indicating the first tax/fee computation formula. Computation formulas are presented in Appendix F
j	tax/fee flag	number(1	N/A	A code indicating a tax/fee is required. Valid Tax/Fee Flag Codes are: 1 = Tax/fee required 2 = Tax/fee may be required. Not used in Merchandising
k	tax/fee specific rate	Number (12)	blank if no value	The specific rate of duty required to compute taxes and/or fees. Eight decimal places are implied
l	tax/fee ad valorem	Number (12)	blank if no value	The ad valorem rate of duty required to compute taxes and/or fees. Eight decimal places are implied

Table 18–5 (Cont.) Input File Layout

Record Name		Field Name	Field Type	Default Value	Description
m		space fill	char(1)	blank	Space fill
VD	a	Control identifier	char(1)	V	identifies start of record
	b	Record type	char(1)	D	identifies record type
	c	tariff number	Number (10)	N/A	unique tariff number
	d	Special Program Indicator (SPI) Code	char(32)	N/A	A code indicating if a tariff number is subject to a special program. Up to sixteen additional 2-position codes can be reported. Left justify. The SPI codes are not reported in any particular sequence
	e	Filler	char(36)	N/A	Space fill

Note: V6 through VC records have the same fields as the V5 record.

Design Assumptions

N/A

tranupld (Transportation Upload)

Module Name	tranupld.pc
Description	Transportation Upload
Functional Area	Oracle Retail Trade Management
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS140
Wrapper Script	rmswrap_multi_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program uploads data from trading partners about the transportation of merchandise from the manufacturing site through customs clearance.

Restart/Recovery

The logical unit of work is a valid DTRAN record. The program reads each DTRAN record from the upload file, validates it and processes it. The recommended commit max counter value for this program is 1000 (this value depends on the implementation).

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000177

Input File Layout

Table 18–6 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FTRAN	Record descriptor	Char(5)	FTRAN	File head marker
	Line id	Number(10)	0000000001	Unique line id
	File type definition	Char(4)	TRUP	Identifies program as tranupld
	File create date	Char(14)	Current date	YYYYMMDD HHMISS format

Table 18–6 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
DTRAN	Record descriptor	Char(5)	DTRAN	Vessel, Voyage, ETD, Container, BL, Invoice File head
	Line id	Number(10)	N/A	Unique line id
	Partner Type	Char(6)	N/A	Identifies the partner type
	Partner ID	Char(10)	N/A	Identifies the partner id
	Vessel ID	Char(20)	N/A	Identifies the Vessel
	Voyage ID	Char(10)	N/A	Identifies the Voyage or Flight ID
	Estimated Depart Date	Char(8)	N/A	YYYYMMDD format
	Shipment Number	Char (20)	N/A	Identifies an outside Shipment number
	Actual Arrival Date	Char(8)	N/A	YYYYMMDD format
	Trans Mode	Char(6)	N/A	Identifies the type of transportation being used. Valid values are found in the TRMO Code Type on the CODE_ DETAIL table
	Vessel SCAC Code	Char(6)	N/A	Customs defined ID for the Vessel. Validated against SCAC table

Table 18–6 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Estimated Arrival Date	Char(8)	N/A	YYYYMMDD format
	Lading Port	Char(5)	N/A	Identifies the Lading Port. Validated against OUTLOC with type = 'LP'
	Discharge Port	Char(5)	N/A	Identifies the Discharge Port. Validated against OUTLOC with type = 'DP'
	Service Contract Number	Char(15)	N/A	Identifies the outside Service Contract Number
	Container id	Char(20)	N/A	Identifies the Container
	Container SCAC code	Char(6)	N/A	Customs defined id for the container. Validated against SCAC table
	Delivery Date	Char(8)	N/A	YYYYMMDD format
	Seal id	Char(15)	N/A	Customs defined id for the container's seal
	Freight Type	Char(6)	N/A	Code that identifies the container type. Validated against the FREIGHT_ TYPE table
	Freight Size	Char(6)	N/A	Code that identifies the container size. Validated against the FREIGHT_ SIZE table
	In Transit No.	Char(15)	N/A	External transit number
	In Transit Date	Char(8)	N/A	YYYYMMDD format

Table 18–6 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	BL/AWB id	Char(30)	N/A	Identifies the Bill of Lading or Air Way Bill
	Candidate Ind	Char(1)	Defaulted to 'N'	Identifies a complete Transportation record. Valid values are 'Y' and 'N'
DPOIT	Record descriptor	Char(5)	DPOIT	Order/Item detail info
	Line id	Number(10)	N/A	Unique file line id
	ACD_Code	Char(1)	N/A	Determines which process to perform 'A'dd, 'C'hange, 'D'elete.
	Rush Ind	Char(1)	Defaulted to 'N'	Identifies whether or not the item should be on a 'Rush' delivery. Valid values are 'Y' and 'N'
	Order number	Number(8)	N/A	Merchandising order number
	Item	Char(25)	N/A	Merchandising Item number
	Invoice id	Char(30)	N/A	Identifies the Commercial Invoice
	Invoice date	Char(8)	N/A	YYYYMMDD format
	Currency Code	Char(3)	N/A	Currency that the Currency Amount is reported in. Validated against CURRENCIES table.
	Exchange Rate	Char (20)	N/A	The exchange rate back to the primary currency (10 implied decimals)

Table 18–6 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Invoice amt	Char (20)	N/A	Invoice amt*10000 (with 4 implied decimal places), amount charged by supplier for the PO/Item
	Origin Country id	Char(3)	N/A	Identifies where the PO/Item was made
	Consolidation Country id	Char(3)	N/A	Identifies where the PO/Items were consolidated
	Export Country id	Char(3)	N/A	Identifies where the PO/Items where shipped from
	Status	Char(6)	N/A	Identifies the PO/Item status. Valid values are found in the TRCO Code Type on CODE_ DETAIL
	Receipt ID	Char(30)	N/A	Identifies the external receipt number
	FCR id	Char(15)	N/A	Identifies the Freight Cargo Receipt id
	FCR date	Char(8)	N/A	YYYYMMDD format
	Packing Method	Char(6)	N/A	Identifies the Packing Type (Hanging or Flat). Valid values are 'HANG' or 'FLAT'
	Lot Number	Char(15)	N/A	Identifies the Lot Number of the PO/Item

Table 18–6 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Item Qty	Number(12)	N/A	Item Qty*10000(with 4 implied decimals), qty of Items
	Item QTY UOM	Char(4)	N/A	Identifies the UOM associated with the item quantity
	Carton QTY	Number(12)	N/A	Carton QTY*10000 (with 4 implied decimals), qty of Cartons
	Carton QTY UOM	Char(4)	N/A	Identifies the UOM associated with the carton quantity
	Gross WT	Number(12)	N/A	Gross WT*10000 (with 4 implied decimals), Gross weight
	Gross WT UOM	Char(4)	N/A	Identifies the UOM associated with the gross weight
	Net WT	Number(12)	N/A	Net WT*10000 (with 4 implied decimals), Net Weight
	Net WT UOM	Char(4)	N/A	Identifies the UOM associated with the net weight
	Cubic	Number(12)	N/A	Cubic*10000 (with 4 implied decimals), cubic size
	Cubic UOM	Char(4)	N/A	Identifies the UOM associated with the cubic size
	Comments	Char(256)	N/A	User Comments

Table 18–6 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FTAIL	Record type	Char(5)	FTAIL	N/A
	Line id	Number(10)	N/A	Unique file line id
	No. of lines	Number(10)	N/A	Total number of transaction lines in file (not including FHEAD and FTAIL)

Design Assumptions

N/A

lcadnld (Letter of Credit Application Download)

Module Name	Lcadnld.pc
Description	Letter of Credit Application Download
Functional Area	Retail Trade Management
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS57
Wrapper Script	rmswrap_dnld_in.ksh

Schedule

See Oracle Merchandising Batch Schedule.

Design Overview

Lcadnld sends letter of credit (LC) applications to partner banks. Online user actions flag LCs for download by writing to the LC_DOWNLOAD table.

Restart/Recovery

Restart/recovery for this program is set up at the lc_ref_id level. The recommended commit counter setting is 10000 records (subject to change based on experimentation).

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integratin Contract	IntCon000052

Output File Layout

Table 18–7 File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Identifier	Number(10)	line number in file	ID of current line being created for output file
	File Type Definition	Char(4)	LCAP	Identifies file as 'Letter of Credit Application'
	File Create Date	Char(14)	create date	Current date, formatted to 'YYYYMMDD HH24MISS'

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Detail	File Type Record Descriptor	Char(5)	THEAD	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	ID of current line being created for output file.
	Transaction Set Control Number	Number(10)	sequence number	Used to force unique file check
	Issuing Bank	Char(10)	lc_ head.issuing _bank	Used to sort the LCs into individualized bank SWIFT formatted files (using another program) - bank where LC application is headed
	Issuing Bank Name	Char(240)	partner.partn er_desc	The description from the partner table where partner_id = issuing_bank and partner_ type = 'BK'
	Issuing Bank Address 1	Char(240)	addr.add_1	Mandatory line of address
	Issuing Bank Address 2	Char(240)	addr.add_2	Non-mandator y line of address (can be null)
	Issuing Bank Address 3	Char(240)	addr.add_3	Non-mandator y line of address (can be null)
	Issuing Bank City	Char(120)	addr.city	City bank located in
	Issuing Bank State	Char(3)	addr.state	State, if applicable, where bank located in
	Issuing Bank Post Code	Char(30)	addr.post	Post code, if applicable, where bank located in
	Issuing Bank Country	Char(3)	addr.country _id	Country bank located in

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Advising Bank	Char(10)	lc_ head.advising_bank	Used to sort the LCs into individualized bank SWIFT formatted files (using another program) - bank where LC application is headed
	Advising Bank Name	Char(240)	Partner.partner_desc	The description from the partner table where partner_id = advising_bank and partner_ type = 'BK'
	Advising Bank Address 1	Char(240)	Addr.add_1	Mandatory line of address
	Advising Bank Address 2	Char(240)	Addr.add_2	Non-mandatory line of address (can be null)
	Advising Bank Address 3	Char(240)	Addr.add_3	Non-mandatory line of address (can be null)
	Advising Bank City	Char(120)	Addr.city	City bank located in
	Advising Bank State	Char(3)	Addr.state	State, if applicable, where bank located in
	Advising Bank Post Code	Char(30)	Addr.post	Post code, if applicable, where bank located in
	Advising Bank Country	Char(3)	Addr.country_id	Country bank located in
	Letter of Credit	Number(8)	lc_head.lc_ ref_id	The LC_REF_ ID off the LC_ HEAD table
	Form Type	Char(6)	lc_ head.form_ type	The level of detail that the LC will send to the issuing bank
	Form Type Description	Char(40)	code_ detail.code_ desc	Describes the form type: Long or Short

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Letter of Credit Type	Char(6)	lc_head.lc_type	Describes the form type: Long or Short
	Letter of Credit Type Description	Char(40)	code_detail.code_desc	Describes the LC type: Master, Normal, Revolving
	Form of Letter of Credit – I	Char(1)	sup_import_attr.revocable_ind	The REVOCABLE_IND from the SUP_IMPORT_ATTR table
	Form of Letter of Credit – II	Char(1)	lc_head.transferable_ind	Indicates if LC transferable
	Application Date	Char(14)	lc_head.application_date	Date the LC is created within Import Management/ Merchandising, formatted to 'YYYYMMDD HH24MISS'
	Expiration Date	Char(14)	lc_head.expiration_date	The date the LC expires, formatted to 'YYYYMMDD HH24MISS'
	Place of Expiry	Char(6)	lc_head.place_of_expiry	Code for the place the LC will expire
	Place of Expiry Description	Char(40)	desc is retrieved through a decode	The description of the place the LC will expire
	Applicant	Char(10)	lc_head.applicant	Party on whose behalf the LC is being issued
	Applicant Name	Char(240)	partner.partner_desc	The description from the partner table where partner_id = applicant and partner_type = 'AP'
	Applicant Address 1	Char(240)	addr.add_1	Mandatory line of address
	Applicant Address 2	Char(240)	addr.add_2	Non-mandatory line of address (can be null)

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Applicant Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Applicant City	Char(120)	addr.city	City applicant located in
	Applicant State	Char(3)	addr.state	State, if applicable, where applicant located in
	Applicant Post Code	Char(10)	addr.post	Post code, if applicable, where applicant located in
	Applicant Country	Char(3)	addr.country_id	Country applicant located in
	Beneficiary	Number(10)	lc.head.beneficiary	Party in favor of which the LC is being issued
	Beneficiary Name	Char(240)	sups.sup_name	Beneficiary (supplier) name from the SUPS table
	Beneficiary Address 1	Char(240)	addr.add_1	Mandatory line of address
	Beneficiary Address 2	Char(240)	addr.add_2	Non-mandatory line of address (can be null)
	Beneficiary Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Beneficiary City	Char(120)	addr.city	City beneficiary located in
	Beneficiary State	Char(3)	addr.state	State, if applicable, where beneficiary located in
	Beneficiary Post Code	Char(30)	addr.post	Post code, if applicable, where beneficiary located in

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Beneficiary Country	Char(3)	addr.country_id	Country beneficiary located in
	Currency Code	Char(3)	lc_head.currency_code	The country of origin for the orders on the LC
	Exchange Rate	Number (20,10)	lc_head.exchange_rate	Exchange_rate to convert LC currency to Merchandising currency
	Origin Country ID	Char(3)	lc_head.origin_country_id	Origin country of the orders associated with the LC
	Presentation Terms	Char(6)	lc_head.presentation_terms	Code for the terms of presentation
	Presentation Terms Description	Char(40)	desc is retrieved through a decode	Description of the terms of presentation
	Purchase Type	Char(6)	lc_head.purchase_type	Code for the purchase type
	Purchase Type Description	Char(40)	desc is retrieved through a decode	Description of the purchase type
	Advice Method	Char(6)	lc_head.advice_method	Code for the advice method
	Advice Method Description	Char(40)	desc is retrieved through a decode	Description of the advice method (eg. Full Wire, Mail, and so on)
	Issuance	Char(6)	lc_head.issuance	Code for the issuance
	Issuance Description	Char(40)	desc is retrieved through a decode	Description of the issuance (for example Cable, Telex, and so on)

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Amount Type	Char(6)	lc_ head.amount _type	If 'E'xact, then amount must be exat, if 'A'pproximate then amount can be within variance percent
	Amount Type Description	Char(40)	desc is retrieved through a decode	Description of amount_type
	Amount	Number (20,4)	lc_ head.amount	The total amt of the Letter of Credit
	Variance Percent	Number (12,4)	lc_ head.varianc e_pct	Allowed currency variance percent for the LC
	Specification	Char(6)	lc_ head.specific ation	Code for any condition for the credit, such as,. "maximum", and so on
	Specification Description	Char(40)	desc is retrieved through a decode	Description of condition for the credit, such as,. "maximum", and so on
	Credit Available With	Char(10)	lc_ head.credit_ avail_with	Code for bank with which credit is available
	Credit With Bank Name	Char(40)	partner.partn er_desc	The description from the partner table where partner_id = credit_avail_ with and partner_type = 'BK'
	Credit With Address 1	Char(240)	addr.add_1	Mandatory line of address
	Credit With Address 2	Char(240)	addr.add_2	Non-mandator y line of address (can be null)

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Credit With Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Credit With City	Char(120)	addr.city	City creditor located in
	Credit With State	Char(3)	addr.state	State, if applicable, where creditor located in
	Credit With Post Code	Char(30)	addr.post	Post code, if applicable, where creditor located in
	Credit With Country	Char(3)	addr.country_id	Country creditor located in
	Drafts At	Char(6)	lc_head.drafts_at	Specifies the terms of the drafts to be drawn under the LC
	Drafts At Description	Char(40)	desc is retrieved through a decode	Description of the terms of the drafts to be drawn under the LC
	Drawee	Char(10)	lc_head.paying_bank	Identifies drawee of drafts to be drawn under LC (paying bank)
	Drawee Name	Char(240)	partner.partner_desc	The description from the partner table where partner_id = paying_bank and partner_type = 'BK
	Drawee Address 1	Char(240)	addr.add_1	Mandatory line of address
	Drawee Address 2	Char(240)	addr.add_2	Non-mandatory line of address (can be null)
	Drawee Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Drawee City	Char(120)	addr.city	City bank located in
	Drawee State	Char(3)	addr.state	State, if applicable, where bank located in
	Drawee Post Code	Char(30)	addr.post	Post code, if applicable, where bank located in
	Drawee Country	Char(3)	addr.country_id	Country bank located in
	Negotiating Bank	Char(10)	lc_head.negotiating_bank	Identifies the negotiating bank
	Negotiating Bank Name	Char(240)	partner.partner_desc	The description from the partner table where partner_id = negotiating_bank and partner_type = 'BK'
	Negotiating Bank Address 1	Char(240)	addr.add_1	Mandatory line of address
	Negotiating Bank Address 2	Char(240)	addr.add_2	Non-mandatory line of address (can be null)
	Negotiating Bank Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Negotiating Bank City	Char(120)	addr.city	City bank located in
	Negotiating Bank State	Char(3)	addr.state	State, if applicable, where bank located in
	Negotiating Bank Post Code	Char(30)	addr.post	Post code, if applicable, where bank located in
	Negotiating Bank Country	Char(3)	addr.country_id	Country bank located in
	Confirming Bank	Char(10)	lc_head.confirming_bank	

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Confirming Bank Name	Char(240)	partner.partner_desc	Identifies the confirming bank
	Confirming Bank Address 1	Char(240)	addr.add_1	The description from the partner table where partner_id = confirming_bank and partner_type = 'BK'
	Confirming Bank Address 2	Char(240)	addr.add_2	Mandatory line of address
	Confirming Bank Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Confirming Bank City	Char(120)	addr.city	Non-mandatory line of address (can be null)
	Confirming Bank State	Char(3)	addr.state	City bank located in
	Confirming Bank Post Code	Char(30)	addr.post	State, if applicable, where bank located in
	Confirming Bank Country	Char(3)	addr.country_id	Post code, if applicable, where bank located in
	Transferring Bank	Char(10)	lc_head.transferring_bank	Country bank located in
	Transferring Bank Name	Char(240)	partner.partner_desc	Identifies the transferring bank
	Transferring Bank Address 1	Char(240)	addr.add_1	The description from the partner table where partner_id = transferring_bank and partner_type = 'BK'
	Transferring Bank Address 2	Char(240)	addr.add_2	Mandatory line of address

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Transferring Bank Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Transferring Bank City	Char(120)	addr.city	Non-mandatory line of address (can be null)
	Transferring Bank State	Char(3)	addr.state	City bank located in
	Transferring Bank Post Code	Char(30)	addr.post	State, if applicable, where bank located in
	Transferring Bank Country	Char(3)	addr.country_id	Post code, if applicable, where bank located in
	Partial Shipment Indicator	Char(1)	lc_head.partial_ship_ind	Country bank located in
	Transshipment Indicator	Char(1)	lc_head.transshipment_ind	Indicates whether goods covered by LC can be partially shipped or not
	Fob Title Pass	Char(6)	lc_head.fob_title_pass	Indicates whether goods can be transferred to another vessel midway through the voyage
	Fob Title Pass Decode	Char(40)	desc is retrieved through a decode	Indicates where the title for goods is passed from the vendor to the purchaser
	Fob Title Pass Description	Char(250)	lc_head.ob_title_pass_desc	Decode of where the title for goods is passed from the vendor to the purchaser
	Transportation to	Char(5)	lc_head.transportation_to	Describes the FOB_TITLE_PASS - could be city name and so on

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Transportation to description	Char(150)	outloc.outloc_desc	Transportation to location
	With Recourse Indicator	Char(1)	lc_head.with_recourse_ind	Description of transportation to location
	Latest Shipment Date	Char(14)	lc_head.latest_ship_date	Indicates conditional payment on the part of the bank as instructed by the buyer
	Earliest Shipment Date	Char(14)	lc_head.earliest_ship_date	Latest ship date for all Pos included in the LC, formatted to 'YYYYMMDD HH24MISS'
	Letter of Credit Negotiation Days	Number(3) replaces x in the string "DOCUMENTS TO BE PRESENTED WITHIN x DAYS AFTER ISSUANCE OF THE SHIPPING DOCUMENTS BUT WITHIN THE VALIDITY OF THIS CREDIT"	lc.head.lc_neg_days	The number of days to negotiate documents
	Bank's LC reference id	Number(8)	lc_head.bank_lc_id	Bank's LC ref id
	File Type Record Descriptor	Char(5)	THDCM	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	ID of current line being created for output file
	Transaction Set Control Number	Number(10)	sequence number	Used to force unique file check

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Header Level Comments	Char(2000)	lc_ head.comme nts	Holds any comments that you added to the Letter of Credit.
	File Type Record Descriptor	Char(5)	TDOCS	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	ID of current line being created for output file
	Transaction Set Control Number	Number(10)	sequence number	Used to force unique file check
	Swift Tag	Char(6)	doc.swift_tag	Identifies individual document types that can be associated with an LC
	Document ID	Number(6)	req_doc.doc_ id	Uniquely identifies the individual documents associated with an LC
	Body Text	Char(2000)	req_doc.doc_ text	Documents associated with a given LC
				Description of Goods and Services OR Documents Required OR Additional Conditions OR Narrative
	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	ID of current line being created for output file
	Transaction Set Control Number	Number(10)	sequence number	Used to force unique file check
	Order Number	Number(8)	lc_ detail.order_ no	PO associated with the LC

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Item	Char(25)	lc_detail.item	Item on the PO - item is rolled up to the item_level of 1, if possible
	Cost	Number (20,4)	lc_detail.cost	If form_type = 'S'hort then cost is the total cost of the order; if the form_type = 'L'ong then the cost is the unit cost of the item
	Quantity	Number (12,4)	lc_detail.qty	Total qty of the item for the order on the LC
	Standard UOM	Char(4)	Item_ master.standard_uom	Standard unit of measure of the quantity of the item for the order on the LC
	Earliest Ship Date	Char(14)	lc_ detail.earliest_ship_date	The earliest date an order on the LC can be shipped, formatted to 'YYYYMMDD HH24MISS
	Latest Ship Date	Char(14)	lc_ detail.latest_ ship_date	The latest date an order on the LC can be shipped, formatted to 'YYYYMMDD HH24MISS'
	item description	Char(250)	Item_ master.desc_ up	Item's description
	File Type Record Descriptor	Char(5)	TMERC	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	ID of current line being created for output file
	Transaction Set Control Number	Number(10)	sequence number	Used to force unique file check

Table 18–7 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Merchandise Description	Char(2000)	lc_ detail.merch _desc	Contains the merchandise description of the field.
	File Type Record Descriptor	Char(5)	TDTCM	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	ID of current line being created for output file
	Transaction Set Control Number	Number(10)	sequence number	Used to force unique file check
	Detail Level Comments	Char(2000)	lc_ detail.comme nts	Holds any comments that you added to the Letter of Credit detail record.
File Trailer	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	ID of current line being created for output file
	Transaction Set Control Number	Number(10)	sequence number	Used to force unique file check
	Transaction detail line count	Number(10)	ID of current line being created for output file	Sum of the detail lines within a transaction
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Identifier	Number(10)	Sequential number Created by program.	ID of current line being created for output file.
	File Record Counter	Number(10)	N/A	Number of records/transa ctions processed in current file (only records between head & tail)

lcmt700 (SWIFT File Conversion - Letter of Credit Application)

Module Name	lcmt700
Description	SWIFT File Conversion – Letter of Credit Application
Functional Area	Oracle Retail Trade Management
Module Type	Integration
Module Technology	Perl
Catalog ID	RMS136
Wrapper Script	rmswrap_perl.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This Perl script will convert the standard Merchandising flat file into the bank specific S.W.I.F.T. MT 700 output files. The input file for this Perl script is the output of the lcadnld.pc Merchandising batch. One output file will be created for each issuing bank in the lcadnld.pc output file.

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000052 (input) IntCon000137 (output)

Output

All files layouts input and output the SWIFT MT 700. The output file should be in the following format:

- Most output fields are contained in their own line (or 3-4 line for addresses).
- Each application consists of four parts, one MT 700 and three MT 701s, which are ordered through the Sequence of Total field: for example, ':27:1/4 MT 700' is the first (MT 700) part of the application.
- MT 700 and MT 701s will be mingled in the same file.
- Each record starts with a colon and a SWIFT field identifier, followed by another colon: for example, ':40A:'-
- Each application is separated by a line with only the ASCII 3 symbol (a heart) on it.

Examples of how individual lines of the MT 700 or MT 701 should look:

```
:27:1/4
:40A:IRREVOCABLE
:20:29893098
:23:NOREF
```

```
:31C:910906
:31D:911022DALLAS
:51D:NORTHERN TRUST INT'L BANKING CORP.
       ONE WORLD TRADE CENTER
SUITE 3941
NY, NY 10048 USA
```

The layout of the S.W.I.F.T MT 700 (Issue of a Documentary Credit) file is as follows:

SWIFT I.D. DATA TYPE CODES (refer to SWIFT User Handbook - Standards general Information - October 1998 release for formatting information):

> **Note:** There is always a new line (nl) after every individual SWIFT ID (and there may be more than one line within an individual field [for example, 59 – Beneficiary, four lines to hold address information]).
>
> In some situations, certain fields will be blank. These fields should be skipped over. In other words, no blank line or tag should be printed indicating the field is blank. Simply ignore it.

lcupld (Letter of Credit Confirmation Upload)

Module Name	lcupld.pc
Description	Letter of Credit Confirmation Upload
Functional Area	Oracle Retail Trade Management
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS55
Wrapper Script	Rmswrap_in_rej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The LCUPLD program is used to upload LC (Letter of Credit) confirmations from bank partners.

After this program has processed a confirmation, the appropriate tables will be updated; a confirmation will update the LC to confirm status and it will write the appropriate records to the LC_ACTIVITY table.

Restart/Recovery

Restart/recovery for this program is set up at the individual FDETL record. Although there may be more than one FDETL record for a given LC, they will each be processed as a separate entity.

File based restart/recovery must be used. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O. The recommended commit counter setting is 10000 records.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integratin Contract	IntCon000054

Input File Layout

Table 18–8 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Sequence Number	Number(10)	0000000001	Line number of the current file
	File Type Definition	Char(4)	LCUP	Identifies file as 'Letter of Credit Upload'
	File Create Date	Char (14)	vdate	Date file was written by external system 'YYYYMMDD HH24MISS' format

Table 18–8 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies file record type
	File Line Sequence Number	Number(10)		Line number of the current file
	Sender's Reference	Char(16)	lc_head.bank_lc_id	The LC number that the bank assigns to a Letter of Credit
	Receiver's Reference	Number(8)	lc_activity.lc_ref_id	The LC number that Trade Management assigned to the Letter of Credit
	Date of Message Being Acknowledged	Char(14)	lc_activity.activity_date	YYYYMMDDHH24MISS format
	Comments	Char(2000)	lc_activity.comments	This field is a concatenation of the following SWIFT fields: 71B – Charges, 72 – Sender information
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	Total number lines	Number(10)	N/A	Total number of lines in file not including FHEAD and FTAIL

lcmt730 (SWIFT File Conversion - Letter of Credit Confirmation)

Module Name lcmt730

Description SWIFT File Conversion – Letter of Credit Confirmation

Functional Area Oracle Retail Trade Management

Module Type Integration

Module Technology	Perl
Catalog ID	RMS138
Wrapper Script	batch_lcmt730.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The lcmt730 Perl script converts letter of credit confirmations from a S.W.I.F.T. format (MT730) to a Merchandising flat file format. The output file from this script will be the input file for the lcupld.pc.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integratin Contract	IntCon000054 (output) IntCon000139 (input)

Input File Layout

Table 18–9 Input File Layout

SWIFT I.D. and Description	Data Type	Description	How MT 730 fields are put into the Merchandising standard file format and what should be the size of Merchandising to be dealt with	Comments
20 - Sender's Reference	16x	LC number. The one assigned by the Sender (issuing bank)	FDETL - Sender's reference, Char(16)	This field maps to Trade Management's Bank LC Ref ID.

Table 18–9 (Cont.) Input File Layout

SWIFT I.D. and Description	Data Type	Description	How MT 730 fields are put into the Merchandising standard file format and what should be the size of Merchandising to be dealt with	Comments
21 - Receiver's Reference	16x	LC number assigned by the Receiver (retailer)	FDETL - Receiver's reference, Number(8) (NOREF used if unknown)	This field maps to Trade Management's LC Ref ID. If this field has 'NOREF', the record must be rejected since this field is used to indicate the LC within Trade Management to which this record applies.
25 - Account Identification	35x	Identifies the number of the account, which has been used for the settlement of charges, on the books of the Sender.	N/A	Trade Management currently does not have fields that map directly to this. Current position - will be included in the input file. However, it will be ignored during the upload process.

Table 18–9 (Cont.) Input File Layout

SWIFT I.D. and Description	Data Type	Description	How MT 730 fields are put into the Merchandising standard file format and what should be the size of Merchandising to be dealt with	Comments
30 - Date of Message Being Acknowledged	6!n	When a message is acknowledging a MT700, this field specifies the date of issue. In all other cases, this field specifies the date on which the message being acknowledged was sent.	FDETL - Date of message Being Acknowledged, Date	This field maps to the LC activity date. As well, if this in confirming an LC application, it will be mapped to the LC's confirmation date. Year interpretation: If YY>79 then YYMMDD = 19YYMMDD Else YYMMDD = 20YYMMDD.
32a - Amount of Charges	Option B - 3!a15d Option D - 6!n3!a15d	Contains the currency code and total amount of charges claimed by the sender of the message. When charges have been debited, D is used (:32D) and when reimbursement for charges is needed, B is used (:32B).	FDETL -Upload_type = 'C'onfirmation	Current position - Because the 730 will only be used for confirmations, this field will not contain any values. The upload type should be set equal to 'C'onfirmation.

Table 18–9 (Cont.) Input File Layout

SWIFT I.D. and Description	Data Type	Description	How MT 730 fields are put into the Merchandising standard file format and what should be the size of Merchandising to be dealt with	Comments
57a - Account With Bank	Option A - [/1!a][/34x] 4!a2!a2!c[3!c] Option D - [/1!a][/34x] 4*35x	This field specifies the bank to which the amount of charges is to be remitted in favor of the Sender.	FDETL - Account With Bank, Char(10)	Current position - will be added to the input file however will be ignored in the upload process. Because Trade Management has no facilities to maintain BICs or party identifiers, option D will always be used for this field (that is, 57D) without [/1!a][/34x] party identifier.
71B - Charges	6*35x	Specification of the charges claimed.	FDETL - Comments, Char(2000)	This field maps to Trade Management's activity comments field. Sender to Receiver information (72) will be concatenated to this.
72 - Sender to Receiver Information	6*35x	Text explanation if wanted.	FDETL - Comments, Char(2000)	This field maps to Trade Management's activity comments field. Charges (71B) will be concatenated to this.

Output File Layout

Table 18–10 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Sequence Number	Number(10)	specified by external system	Line number of the current file
	File Type Definition	Char(4)	LCUP	Identifies file as 'Letter of Credit Upload'
	File Create Date	Char (14)	vdate	date file was written by external system 'YYYYMMDD HH24MISS' format

Table 18–10 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies file record type
	File Line Sequence Number	Number(10)	specified by external system	Line number of the current file
	Sender's Reference	Char(16)	lc_head.bank_ld_id	The LC number that the bank assigns to a Letter of Credit
	Receiver's Reference	Number(8)	lc_activity.lc_ref_id	The LC number that Merchandising assigned to the Letter of Credit
	Date of Message Being Acknowledged	Date (char 8)	lc_activity.activity_date	If the upload type is 'L' then this date will match the date MT 700 date of issue (which we have not resolved between being the vdate or the lc_head.application_date) 'YYYYMMDD' format
	Comments	Char(2000)	lc_activity.comments	Need to truncate? This field will probably be a concatenation of the following SWIFT fields: 71B – Charges, 72 – Sender information
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Sequence	Number(10)	Specified by external system	Line number of the current file
	Total number of lines	Number(10)	Specified by external system	Total number lines in file

Design Assumptions

N/A

lcmdnld (Letter of Credit Amendment Download)

Module Name	lcmdnld.pc
Description	Letter of Credit Amendment Download
Functional Area	Oracle Retail Trade Management
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS56
Wrapper Script	rmswrap_dnld_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

lcmdnld.pc downloads amended letter of credit information to a bank, in the S.W.I.F.T. format.

Online user actions flag LCs for download by writing to the LC_DOWNLOAD table.

Restart/Recovery

Restart/recovery for this program is set up at the lc_ref_id level. The recommended commit counter setting is 1000 records (subject to change based on experimentation).

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integratin Contract	IntCon000053

Output File Layout

Table 18–11 File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Sequence Number	Number(10)	Line number in file	Keeps track of the record's position in the file by line number
	File Type Definition	Char(4)	LCAM	Identifies file as 'Letter of Credit Amendment'
	File Create Date	Char(14)	Create date	Current date, formatted to 'YYYYMMDDHH24MISS'
Transaction Header	Filetype Record descriptor	Char(5)	THEAD	Identifies file record type
	File Line Sequence Number	Number (10)	Line number in file	Keeps track of the record's position in the file by line number
	Transaction Set Control Number	Number (10)	Sequence number	Used to force unique file check

Table 18–11 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Issuing Bank	Char(10)	lc_ head.issuing _bank	Used to sort the LCs into individualized bank SWIFT formatted files (using another program) – bank where LC application is headed
	Issuing Bank Name	Char(240)	partner.partn er_desc	The description from the partner table where partner_id = issuing_bank and partner_type = 'BK'
	Issuing Bank Address 1	Char(240)	addr.add_1	Mandatory line of address
	Issuing Bank Address 2	Char(240)	addr.add_2	Non-mandatory line of address (can be null)
	Issuing Bank Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Issuing Bank City	Char(120)	addr.city	City bank located in
	Issuing Bank State	Char(3)	addr.state	State, if applicable, where bank located in
	Issuing Bank Post Code	Char(30)	addr.post	Post code, if applicable, where bank located in
	Issuing Bank Country	Char(3)	addr.country _id	Country bank located in
	Letter of Credit	Number (8)	lc_detail.lc_ ref_id	The LC_REF_ID off the LC_ DETAIL table
	Bank Letter of Credit ID	Char(16)	lc_ head.bank_ lc_id	The BANK_LC_ID off the LC_HEAD table
	Currency Code	Char(3)	lc_ head.currenc y_code	The CURRENCY_CODE off the LC_HEAD table
	Date of Issue/ Transfer of the Credit	Char(14)	lc_ head.confirm ed_date	Date the Issuing Bank thinks is the date of issue–when it was officially confirmed, formatted to 'YYYYMMDDHH24MISS'

Table 18–11 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Current Amount of LC	Number (20,4)	N/A	This amount will be calculated in the get_current_amount() function and will be the net amount of the LC calculated only using amendments that have been downloaded. Normally, the net amount is calculated using amendments in the 'D'ownloaded status
	Beneficiary	Number (10)	lc.head.beneficiary	Party in favor of which the LC is being issued
	Beneficiary Name	Char(240)	sups.sup_name	Beneficiary (supplier) name from the SUPS table
	Beneficiary Address 1	Char(240)	addr.add_1	Mandatory line of address
	Beneficiary Address 2	Char(240)	addr.add_2	Non-mandatory line of address (can be null)
	Beneficiary Address 3	Char(240)	addr.add_3	Non-mandatory line of address (can be null)
	Beneficiary City	Char(120)	addr.city	City beneficiary located in
	Beneficiary State	Char(3)	addr.state	State, if applicable, where beneficiary located in
	Beneficiary Post Code	Char(30)	addr.post	Post code, if applicable, where beneficiary located in
	Beneficiary Country	Char(3)	addr.country_id	Country beneficiary located in
Transaction Detail	File Type Record Descriptor	Char(5)	TDETL	Identifies file record type
	File Line Sequence Number	Number (10)	line number in file	Keeps track of the record's position in the file by line number
	Transaction Set Control Number	Number (10)	sequence number	Used to force unique file check

Table 18–11 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Amendment Number	Number (8)	lc_amendments.amend_no	Holds the amendment number for the amendment
	Order_no	Number (8)	lc_amendments.order_no	Order_no, if applicable, that is attached to the LC that is being amended
	Item	Char(25)	lc_amendments.item	Item being amended, either a Style or Staple sku
	Value Being Amended	Char(6)	lc_amendments.amended_value	LC Field being amended. Can be any of the following code_types: CODE CODE_DESC AI Add Item AO Add PO ARQD Add Reqd Doc. C Cost ED Expiration Date ESD Earliest Ship Date LSD Latest Ship Date NA Net Amount ND Negotiation Days OC Origin Country OQ Order Quantity PE Place of Expiry PRT Presentation Terms PSF Partial Ship Flag RI Remove Item RO Remove PO RRQD Remove Reqd Doc TFF Transferable Flag TSF Transshipment Flag

Table 18–11 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Value Being Amended Description	Char(40)	code_ detail.code_ desc	The Value Being Amended decoded (see the above list). Will possibly be used when printing to the SWIFT file MT 707 for clarity
	Original Value of Amended Field	Char(45)	lc_ amendments. original_ value	Current value of field that is being amended
	New Value of Amended Field	Char (2000)	lc_ amendments. new_value	New value of the field that is being amended
	Description of New Value	Char(40)	code_ detail.code_ desc	The new value decoded (or fetched from a table, as in the origin_country case)– only applicable to the following amended values: place of expiry, title_pass_ location, origin_country, presentation terms, purchase type
	Sign	Char(1)	N/A	If the effect is negative it will be "-" if the effect is positive it will be " "
	Effect	Number (20,4)	lc.amendmen ts.effect	Effect that amendment will have on LC if amendment to change qty or cost of a PO or amount of LC itself
	Date of Amendment	Char(14)	Lc_ amendments. accept_date	Date on which Issuing Bank (or issuing party, in this case the retailer) considers the credit as being amended, formatted to 'YYYYMMDD HH24MISS'
Transaction Text	File Type Record Descriptor	Char(5)	TTEXT	Identifies file record type
	File Line Sequence Number	Number (10)	line number in file	Keeps track of the record's position in the file by line number
	Transaction Set Control Number	Number (10)	sequence number	Used to force unique file check
	Amendment Text	Char (2000)	text description	A text description of the individual amendment (for each TDETL line of the output file) built by the package LC_AMEND_SQL. AMEND_TEXT.

Table 18–11 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
Transaction Trailer	File Type Record Descriptor	Char (5)	TTAIL	Identifies File Record Type
	File Line Sequence Number	Number (10)	Line Number in file	ID of current line being created for output file
	Transaction set control number	Number (10)	Sequence number	Used to force unique file check
	Transaction detail line count	Number (10)	ID of current line being created for output file	Sume of the detail lines within a transaction
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Sequence Number	Number (10)	line number in file	Keeps track of the record's position in the file by line number
	Control Number File Line Count	Number (10)	total detail lines	Sum of all transaction lines, not including the file header and trailer

lcmt707 (SWIFT File Conversion – Letter of Credit Amendment)

Module Name lcmt707

Description SWIFT File Conversion – Letter of Credit Amendment

Functional Area Oracle Retail Trade Management

Module Type Integration

Module Technology Perl

Catalog ID RMS137

Wrapper Script rmswrap_perl.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This Perl script converts the Oracle retail standard interface file format for Amendments to Letters of Credit download to the corresponding S.W.I.F.T file format (MT 707). The input file for this Perl script is the output of the lcmdnld.pc Merchandising batch.

I/O Specification

Integration Type	Download to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000053 (input) IntCon000138 (output)

Output

The SWIFT MT 707 output file should be in the following format:

- Most output fields are contained in their own line (or 3-4 line for addresses).
- Each amendment consists of only one part, the MT 707. There may be several MT 707s at any given time associated to an LC because they are grouped by amendment number at the time of creation. All TDETL records with the same amend_no will be grouped together in one MT 707.
- Each record starts with a colon and a SWIFT field identifier, followed by another colon: for example, ':40A:'-
- Each amendment is separated by a line with only the ASCII 3 symbol (a heart) on it.

Logic Setup:

The input file will be in standard Merchandising file format. It will potentially have numerous TDETL lines per each THEAD line. There may be numerous TDETL records for one amendment. MT 707 will write one record for each amendment, so if there are multiple TDETL records they need to be combined. There is one TTEXT for each TDETL.

There are three values that need to be calculated. 32B, 33B, 34B. 32B is the total increment or the sum of the positive effect values for each amendment. 33B is the total decrement or the sum of all the negative effect values for each amendment. 32B and 33B are separate totals for each amendment. 34B is the total difference, so it is the sum of the total increment and total decrement. 34B is not just for one amendment though; it is for all amendments of a THEAD record, so this total will run through each TDETL in a THEAD.

For example: if the input file contains:

- THEAD
- TDETL amendment 1, effect +1000
- TTEXT
- TDETL amendment 1, effect +500
- TTEXT
- TDETL amendment 2, effect -2500
- TTEXT
- TDETL amendment 3, effect +4000
- TTEXT
- TDETL amendment 3, effect -1000
- TTEXT

- TDETL amendment 3, effect +500
- TTEXT
- TDETL amendment 4, effect -1000
- TTEXT
- TDETL amendment 4 , effect –2500
- TTEXT
- TTAIL

```
32B for amendment 1 = 1500
33B for amendment 1 = 0
34B for amendemnt 1 = 1500

32B for amendment 2 = 0
33B for amendment 2 = 2500
34B for amendemnt 2 = -1000

32B for amendment 3 = 4500
33B for amendment 3 = 1000
34B for amendemnt 3 = 4500

32B for amendment 4 = 0
33B for amendment 4 = 3500
34B for amendemnt 4 = 1000
```

Examples of how individual lines of the M T 707 should look:

```
APPLICANT:
OPERATOR:
OPERATION DATE:
OPERATION TIME:
TEST KEY:
BATCH TOTAL:
SEGMENT TOTAL:
MT/PRIORITY:707 02
:27:1/1
:20:10001981
:21:1981
:52D:Bank One
100 Bank One Way
Columbus        ,OH  41984      US
:31C:990204
:30:990204
:26E:1
:59:David Fashion Creations P/L Pack
Wholesale Division
109 Ackland St.
St. Kilda      ,VA  30280-1234 US
:32B:USD500,0
:33B:USD0,0
:34B:USD500,0
:79:Letter of Credit:  has been changed from 25 to 30
for Style 10049369, resulting in an effect of 500
(USD).
```

The layout of the S.W.I.F.T MT 707 (Amendment to a Documentary Credit) file is as follows:

SWIFT I.D. DATA TYPE CODES (refer to SWIFT User Handbook – Standards General Information – October 1998 release for formatting information):

Note: The field lengths and types in the Oracle Retail Standard Download Format of the MT 707 are important because sometimes they are different from the information that is being placed in them and the fields may have to be truncated, rounded, and so on.

There is always a new line (nl) after every individual SWIFT ID (and there may be more than one line within an individual field (example 59 - Beneficiary, four lines to hold address information).

In some situations, certain fields will be blank. These fields should be skipped over. In other words, no blank line or tag should be printed indicating the field is blank. Simply ignore it.

---

lcup798 (Letter of Credit Drawdowns and Charges)

Module Name	lcup798.pc
Description	Letter of Credit Drawdowns and Charges
Functional Area	Oracle Retail Trade Management
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS54
Wrapper Script	rmswrap_in_rej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program reads data from an input file containing letter of credit charges and drawings (in standard Oracle Retail format, modified from the SWIFT 798 format by the lcmt798 Perl script), validates it, and inserts it into the LC_ACTIVITY table. If a record fails validation, it will be written to a reject file. These rejected records can be reprocessed by lcup798 after errors have been corrected.

Restart/Recovery

This program will be restartable but not threadable.

Restart/recovery logic for file-based processing is used. Records will be committed to the database when commit_max_ctr defined in the RESTART_CONTROL table is reached.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter

Integratin Contract	IntCon000055

The input file for this batch program is the output from the lcmt798 Perl script.

Input File Layout

Table 18–12 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File head descriptor	Char(5)	FHEAD	Describes file line type
	Line id	Number (10)	0000000001	Sequential file line number
	File Type Definition	Char(4)	'LCCH'	Identifies as an LC 798 file-Letter of Credit Charges
	Current date	Date	N/A	File date in YYYYMMDD HH24MISS format

Table 18–12 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FDETL	File record descriptor	Char(5)	FDETL	Describes file line type
	Line id	Number (10)		Sequential file line number
	Bank letter of credit reference ID	Char (16)	SWIFT tag 20	Bank's LC ref ID
	Order number	Number(8)	SWIFT tag 21	Order number attached to LC.May be blank
	Invoice number	Number (15)	SWIFT tag 23	NOT a Merchandising invoice number, just a reference invoice number from the issuing bank. May be blank
	Transaction number	Number (10)	N/A	Amendment number or transaction number assigned by bank.May be null
	Transaction code	Char(6)	B or D	'B'ank charge or'D'rawdown
	Amount	Number(21)	SWIFT tag 33A,71A	(This is a 20-digit number with a leading – sign or blank and 4 implied decimal places.) Amount of charge or drawdown
	Currency code	Char(3)	SWIFT 33A,71A	Currency that the amount is in
	Activity date	Date	SWIFT 33A,32C,32D	Activity date(formatted as 'YYYYMMDD')
	Comments	Char(2000)	SWIFT tag 72	Any comments associated with activity.May be null

Table 18–12 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FTAIL	File record descriptor	Char(5)	FTAIL	Marks end of file
	Line id	Char(10)	N/A	Sequential file line number
	Number of lines	Number(10)	N/A	Number of lines in file not counting FHEAD and FTAIL

lcmt798 (SWIFT File Conversion – Letter of Credit Charges and Drawdowns)

Module Name	lcmt798
Description	SWIFT File Conversion – Letter of Credit Drawdowns and Charges
Functional Area	Retail Trade Management - Letter of Credit Interfaces
Module Type	Integration
Module Technology	Perl
Catalog ID	RMS139
Wrapper Script	batch_lcmt798.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This Perl script converts letter of credit (L/C) activity data for charges and drawdowns from a S.W.I.F.T. format input file to a Merchandising format file.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000139 (input)

Input File Layout

Table 18–13 Input File Layout

Swift Tag	Description	Regd?	Datatype	Merchandising Field
20 - Transaction Reference Number	The sender's unambiguous identification of the transaction. Its detailed form and content are at the discretion of the sender.	Yes	16x - Transaction Reference Number	Bank L/C ID Lc_head.bank_lc_id Varchar2(16)
12 - Type of Financial Instrument	This field classifies the financial instrument by a description or proprietary code.	Yes	Option A- :4!c/[8c]/30x :4!c - Qualifier / - Delimiter [8c] - Issuer Code / - Delimiter 30x - Type	This field will contain a constant identifier - '798'
77E - Proprietary Message	This field contains the proprietary message in a format agreed to by the Sender and the Receiver.	Yes	Option E- 73x [n*78x]	This field will contain the information below (fields 21, 23, 32C, 32D, 71A, 33A, 72) Carriage return, Line feed, Colon 'CrLf:' will be used to separate fields included in this 77E For example: :77E:'CrLf' :21:10004321:CrLf' :32C:990121USD1045 and so on. There may be multiple 77Es in one file

Table 18–13 (Cont.) Input File Layout

Swift Tag	Description	Regd?	Datatype	Merchandising Field
21 - Related Reference	This field specifies, in an unambiguous way, a message or transaction identifier which is normally included as part of the information supplied with the message or transaction itself, and can subsequently be used to distinguish the message or transaction identified from other messages or transactions.	No	16x	P/O Number Lc_ activity.order_ no Number(8)
23 - Further identification	This field specifies the type of transaction being confirmed, as well as the settlement method used.	No	16x	Invoice Number Lc_ activity.invoice _no Varchar2(15)

Table 18–13 (Cont.) Input File Layout

Swift Tag	Description	Regd?	Datatype	Merchandising Field
32C - Date and Amount	This field specifies the currency code and amount in a transaction and a corresponding date.	No	Option A- :4!c/[8c]/30x :4!c - Qualifier / - Delimiter [8c] - Issuer Code / - Delimiter 30x - Type	Charges Credited (this is interpreted as a positive amount) Date will be in format YYMMDD The integer part of the Amount must contain at least one digit. A decimal comma ',' is mandatory and is included in the maximum length Lc_activity.amount Number(20,4) Lc_activity.currency_code Varchar2(3) Lc_activity.activity_date Date

Table 18–13 (Cont.) Input File Layout

Swift Tag	Description	Regd?	Datatype	Merchandising Field
32D - Date and Amount	This field specifies the currency code and amount in a transaction and a corresponding date.	No	Option D- 6!n3!a15d 6!n - Date 3!a - Currency 15d - Amount	Charges Debited (this is interpreted as a negative amount) Date will be in format YYMMDD The integer part of the Amount must contain at least one digit. A decimal comma ',' is mandatory and is included in the maximum length Lc_activity.amount Number(20,4) Lc_activity.currency_code Varchar2(3) Lc_activity.activity_date Date

Table 18–13 (Cont.) Input File Layout

Swift Tag	Description	Regd?	Datatype	Merchandising Field
33A - Date and Amount	This field specifies the currency code and amount in a transaction and a corresponding date.	No	Option A- 6!n3!a15d 6!n - Date 3!a - Currency 15d - Amoun	Date, currency, amount of drawing (this is interpreted as a positive amount) Date will be in format YYMMDD The integer part of the Amount must contain at least one digit. A decimal comma ',' is mandatory and is included in the maximum length Lc_ activity.amount Number(20,4) Lc_ activity.currency_code Varchar2(3) Lc_ activity.activity_date Date

Table 18–13 (Cont.) Input File Layout

Swift Tag	Description	Regd?	Datatype	Merchandising Field
33C - Date and Amount	This field specifies the currency code and amount in a transaction and a corresponding date.	No	Option A- 6!n3!a15d 6!n - Date 3!a - Currency 15d - Amount	Date, currency, amount of drawing (this is interpreted as a negative amount) Date will be in format YYMMDD The integer part of the Amount must contain at least one digit. A decimal comma ',' is mandatory and is included in the maximum length. Lc_activity.amount Number(20,4) Lc_activity.currency_code Varchar2(3) Lc_activity.activity_date Date
72 - Sender to Receiver Information	This field specifies instructions or additional information for the Receiver, Intermediary, Account with Institution or Beneficiary Institution.	No	6*35x	Comments Lc_activity.comment Varchar2(2000)
18A - Number of Repetitive Parts	This field specifies the number of times the repetitive part(s)/sequence(s)directly before or after this field appears in the message.	No	Option A- 5n - Number of Repetitive Parts.	Number of 77E's contained within the file.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integratin Contract	IntCon000055 (input)

Output File Layout

Table 18–14 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Identifier	Number (10)	Line number in file	ID of current line being created for output file
	File Type Definition	Char(4)	LCCH	Identifies file as 'Letter of Credit Changes'
	File Create Date	Char(14)	Create date	Current date, formatted to 'YYYYMMDDHH24 MISS'
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies file record type
	File Line Sequence Number	Number (10)	Line number in file	ID of current line being created for output file
	Bank Letter of Credit Reference ID	Char(16)	SWIFT tag 20	Bank L/C ID
	Order Number	Number (8)	SWIFT tag 21	Contains the order number that is attached to the letter of credit
	Invoice Number	Char (15)	SWIFT tag 23	Identifies the Issuing Bank's invoice number to which the drawdown refers. This field does not correspond to a Merchandising invoice number
	Transaction Number	Char (10)	Null	Identifies the amendment number or actual transaction number assigned by the bank

Table 18–14 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Transaction Code	Char (6)	If the transaction is a Bank Charge – 'B' f the transaction is a Drawdown – 'D'	Identifies the type of transaction that occurred The type is determined by what detail fields are received for the record. If the record contains a 33A this field will get a 'D'. If the record contains either a 32C or 32D this field will get a 'B'
	Amount Sign	Char (1)	SWIFT 33A, 33C SWIFT 32C, 32D	If the record contains a 33A field leave a blank space in this field If the record contains a 33C filed this field should contain a '-' If the record contains a 32C field leave a blank space in this field If the record contains a 32D field this field should contain a '-'
	Amount	Number (20)	SWIFT 33A, 33C SWIFT 32C, 32D	Holds the amount of the activity. This field will have 4 implied decimal places If SWIFT 32C or 32D (Bank Charge) contains a value, use the amount from this field If SWIFT 33A or 33C (Drawdown) contains a value, use the amount from this field

Table 18–14 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Currency Code	Char (3)	SWIFT 33A, SWIFT 32C, 32D	Contains the activity's currency code If SWIFT 32C or 32D (Bank Charge) contains a value, use the currency from this field If SWIFT 33A (Drawdown) contains a value, use the currency from this field
	Activity Date	Char (8)	SWIFT 33A, SWIFT 32C, 32D	Holds the date that the activity took place. Formatted to 'YYYYMMDD' If SWIFT 32C or 32D (Bank Charge) contains a value, use the date from this field If SWIFT 33A (Drawdown) contains a value, use the date from this field
	Comments	Char (2000)	SWIFT tag 72	Holds any comments for the activity
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Identifier	Number (10)	Sequential number Created by program.	ID of current line being created for output file
	File Record Counter	Number (10)	N/A	This will contain the number of FDETL lines processed

19

Stock Ledger

The stock ledger holds financial data that allows you to monitor your company's performance. It incorporates financial transactions related to merchandising activities, including sales, purchases, transfers, and markdowns; and is calculated weekly or monthly. The stock ledger accounts for inventory in buckets (how much inventory was returned, how much damaged, and so on). This overview describes how the stock ledger is set up, the accounting methods that impact stock ledger calculations, the primary stock ledger tables, and the batch programs and PL/SQL packages that process data held on the tables.

Note: For additional information about stock ledger transaction posting, see the chapter "Sales Posting" in this volume of the *Oracle Retail Merchandising Operations Guide*.

For additional information about integration of data (including month level stock ledger data) to the General Ledger, see the chapter "Integration - General Ledger' in this volume of the *Oracle Retail Merchandising Operations Guide*

Stock Ledger Set Up and Accounting Methods

The operation of the stock ledger is dependent upon a number of options that you choose for your implementation of Merchandising. To understand how your company uses the stock ledger, you can examine the settings that are described here.

The stock ledger is implemented at the subclass level and supports both the retail and cost methods of accounting. The method of accounting may vary by department and is set on the department (DEPS) table in the profit_calc_type column. The '1' setting indicates that profit is calculated by direct cost. The '2' setting indicates that profit is calculated by retail inventory.

If you select the cost method of accounting, two options are available: average cost or standard cost. The chosen option is represented on the SYSTEM_OPTIONS table in the std_av_ind column, where the standard cost option is indicated by the 'S' setting, and the average cost option is indicated by the 'A' setting. The selected option then applies to all departments that use the cost method stock ledger option.

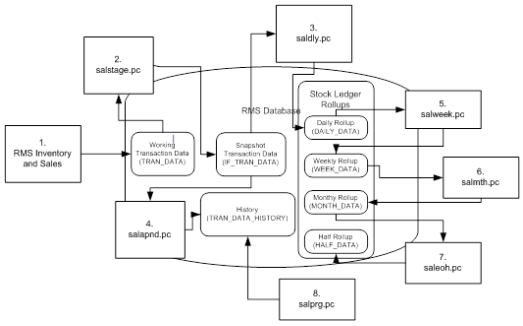
If you select the retail method of accounting, you can choose to implement the retail components of all transactions either to include value-added tax (VAT) or to exclude VAT. You accomplish this through a system-level option vat_ind on the SYSTEM_ OPTIONS table.

Note: If the value-added tax (VAT) system option is enabled in Merchandising, rolled-up stock ledger data values for the retail accounting method include value-added tax.

For sales history purposes, history is maintained based on the calendar that you choose. If your company uses the 4-5-4 calendar, sales history is tracked weekly. If you use the Gregorian (or 'normal') calendar, sales history is tracked monthly. The calendar setting is held on the SYSTEM_OPTIONS table in the calendar_454_ind column.

Process Flow

Figure 19–1 Process Flow - Stock Ledger



1. Assorted Merchandising Inventory and Sales Transactions write to the working transaction data table (TRAN_DATA).
2. Salstage.pc moves transaction data from the working table to the snapshot transaction data table (IF_TRAN_DATA) for additional processing.
3. Saldly.pc rolls up the snapshot transaction data (IF_TRAN_DATA) and persists it to the daily rollup table (DAILY_DATA).
4. Salapnd.pc moves data from the snapshot transaction data table (IF_TRAN_DATA) to the history table (TRAN_DATA_HISTORY).
5. Salweek.pc rolls up daily stock ledger data (DAILY_DATA) to weekly stock ledger data (WEEK_DATA).
6. Salmth.pc rolls up weekly stock ledger data (WEEK_DATA) to monthly stock ledger data (MONTH_DATA).
7. Saleoh.pc rolls up monthly stock ledger data (MONTH_DATA) to half level stock ledger data (HALF_DATA).
8. Salprg.pc deletes aged transaction history (TRAN_DATA_HISTORY).

Batch Design Summary

The following batch designs are included in this functional area:

- salstage.pc (Stage Stock Ledger Transactions for Additional Processing)
- salapnd.pc (Append Stock Ledger Information to History Tables)
- saldly.pc (Daily Rollup of Transaction Data for Stock Ledger)

- salweek.pc (Weekly Rollup of Data/Calculations for Stock Ledger)
- salmth.pc (Monthly Rollup of Data/Calculations for Stock Ledger)
- salmaint.pc (Stock Ledger Table Maintenance)
- stock_ledger_purge_job (Stock Ledger Table Maintenance)
- saleoh.pc (End Of Half Rollup of Data/Calculations for Stock Ledger)
- salprg.pc (Purge Stock Ledger History)
- stkledgr_hist_purge_job (Purge Stock Ledger History)
- nwppurge.pc (Optional End of Year Inventory Position Purge)
- nwp_purge_job (Purge of Aged End of Year Inventory Positions)
- nwpyearend.pc (Optional End of Year Inventory Position Snapshot)
- stlgdnld (Daily or Weekly Download of Stock Ledger Data)
- Otbdlsal (Open To Buy Download Stock Ledger)
- trandataload.ksh (External Transaction Data Upload)
- trandataprocess.ksh (External Transaction Data Process)

salstage (Stage Stock Ledger Transactions for Additional Processing)

Module Name	salstage.pc
Description	Stage Stock Ledger Transactions for Additional Processing
Functional Area	Stock Ledger
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS345
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

In order to make the rollup and extraction of the stock ledger transaction data flexible, this program moves the data on the TRAN_DATA to the IF_TRAN_DATA staging table. This will enable the processes that are writing records to TRAN_DATA to continue in a seamless manner, whereas the processes that rolls the data up to a different level or extract the data to external systems can work without affecting batch timetables.

This process will be achieved by locking the TRAN_DATA table and moving all of the data to the staging table. The original TRAN_DATA table will be emptied and the lock on the table will be released. Before this processing occurs, the staging table will first be emptied to ensure that data is not processed twice. Because the data on the TRAN_DATA and IF_TRAN_DATA tables is very transitional, these tables will fill up and be truncated at least once a day if not several times per day.

Restart/Recovery

N/A

Design Assumptions

N/A

salapnd (Append Stock Ledger Information to History Tables)

Module Name	salapnd.pc
Description	Append Stock Ledger Information to History Tables
Functional Area	Stock Ledger
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS335
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this program is to move data from the staging table for transaction data into the historical transaction data table. This requires placing a lock on the staging table to ensure that no new data will be added to it while the movement is occurring (to handle trickling or real-time processing), moving the data to the historical table, and finally truncating the data from the staging table.

Restart/Recovery

N/A

Design Assumptions

N/A

saldly (Daily Rollup of Transaction Data for Stock Ledger)

Module Name	saldly.pc
Description	Daily Rollup of Transaction Data for Stock Ledger
Functional Area	Stock Ledger
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS336

Wrapper Script rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is responsible for performing the daily summarization processing in the stock ledger in which transaction-level records are fetched from the transaction-level staging table and summed to the subclass/location/day/currency level. Once the records are summarized, they are written to the DAILY_DATA table in both primary as well as the local currency. If the local currency is same as the primary currency, the program will insert records only in local currency.

To call this program the end of day process for the stock ledger would not be completely correct, however, because a day does not really 'close' in the stock ledger until the month closes. Each time that the Daily Stock Ledger Processing program runs, all transaction-level data is processed, whether it is for the current date, a date since the last month closing or even a date prior to the last month closing. For transactions occurring on the current date or since the last month close, they are processed by simply summarizing the date and updating the current information on DAILY_DATA for the date of the transaction. However, if a transaction occurred prior to the last month that was closed (for example:. the transaction was dated 3/15 and the last end of month date was 3/20), then that transaction will be dated with the current date and summarized with the current date's records. Also, in this last case, a warning message will be written to the batch log that alerts you to the problem. The message you will receive is "*ALERT* Transactions have been found for previous months." The sadly post program identifies dept/class/subclass/location combinations within the transactions created during the day which are not available in week and month data tables. These combinations are seeded into the week and month data tables to ensure seamless roll up in the stock ledger.

Restart/Recovery

The logical unit of work is department/class/subclass. This batch program is multithreaded using the v_restart_dept view.

Design Assumption

N/A

salweek (Weekly Rollup of Data/Calculations for Stock Ledger)

Module Name	salweek.pc
Description	Weekly Rollup of Data/Calculations for Stock Ledger
Functional Area	Stock Ledger
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS346
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule.

Design Overview

This program is responsible for performing the weekly summarization processing in the stock ledger. This program processes all weeks that are in the month for which month-end process has not been run, up to the current week. It rolls up data on DAILY_DATA, DAILY_DATA_TEMP and WEEK_DATA_TEMP to the corresponding dept/class/subclass/location/half-month/week/currency level and updates the WEEK_DATA table.

This program processes all weeks that are in the month for which month-end process has not been run, up to the current week. This program can be run at any time during the week - not necessarily just at week-end, as it must be run before the Monthly Stock Ledger Processing, which can be run at any time after the closing of a month.

In addition to the summarization processes done by this program, there are several week ending calculations done as well. The closing stock value, half to date goods available for sale (HTD GAFS), shrinkage and gross margin are calculated by calling a package function, based on the accounting method designated for the department - cost or retail. Additionally, the closing stock value for a processed week becomes opening stock value for the next week. Also, if this program is run at the end of the week, it will write a 'shell' record for the next week, populating the key fields on the table (subclass, location, and so on), the opening stock values at cost and retail and the HTD GAFS at cost and retail. It may be noted that these shell records will be created only for those subclass/location/ week combinations that have a non-zero value of ending inventory or a non-zero value of HTD GAFS.

Restart/Recovery

The logical unit of work is dept/class/subclass combination. A commit will take place when number of dept/class/subclass combination records processed is equal to commit max counter in restart control table.

Design Assumptions

N/A

salmth (Monthly Rollup of Data/Calculations for Stock Ledger)

Module Name	salmth.pc
Description	Monthly Rollup of Data/Calculations for Stock Ledger
Functional Area	Stock Ledger
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS343
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Monthly Stock Ledger Processing program is responsible for performing the monthly summarization processing in the stock ledger in which day-level records are fetched from the transaction-level staging table and summed to the subclass/location/month level. Once the records are summarized, they are written to the MONTH_DATA table. This program processes one month for each program run - starting the latest month to be closed. For example, if it is currently June and both April and May are open, when the program runs, then only April will be closed.

In addition to the summarization processes done by this program, there are several month ending calculations done as well. The closing stock value, half to date goods available for sale (HTD GAFS), shrinkage and gross margin are calculated by calling a package function, based on the accounting method designated for the department - cost or retail. Additionally, the closing stock value for a processed month becomes opening stock value for the next month. Also, when this program is run, it will write a 'shell' record for the next month, populating the key fields on the table (subclass, location, and so on), the opening stock values at cost and retail, the inter-stock take sales and shrinkage amounts and the HTD GAFS at cost and retail. It may be noted that these shell records will be created only for those subclass/location/month combinations that have a non-zero value of either ending inventory, HTD GAFS or inter-stock take amounts.

This program can be run at any time during the month - not necessarily just at month-end. Open stock counts from the month may exist based on the system parameter (CLOSE_MTH_WITH_OPN_CNT_IND). If this indicator is 'Y', then retailers are able to keep a count open across a single month closing in the stock ledger and still close the month financially. A Unit & Value stock count is considered as open until all variances (both unit and value) have been reviewed and applied. Special processing exists if it is allowed and there are open stock counts from the current month. Open stock counts from previous months however cannot exist regardless of the setting.

Restart/Recovery

The logical unit of work (LUW) for this batch program is a dept/class/subclass/loc_type/location/currency_ind record. This batch program is threaded by department using the v_restart_dept view. Processed records are committed to the database after the LUW count has reached the commit_max_ctr.

Design Assumptions

N/A

salmaint (Stock Ledger Table Maintenance)

Module Name	salmaint.pc
Description	Stock Ledger Table Maintenance
Functional Area	Stock Ledger
Module Type	Admin

Module Technology	ProC
Catalog ID	RMS342
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module calls a function to drop partitions on HALF_DATA, DAILY_DATA, WEEK_DATA and MONTH_DATA tables.

Restart/Recovery

N/A

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

I/O Specification

N/A

stock_ledger_purge_job (Stock Ledger Table Maintenance)

Module Name	stock_ledger_purge_job
Description	Stock Ledger Table Maintenance
Functional Area	Stock Ledger
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of one step processing only. It will retain the business logic processing from the original batch program algorithm.

The Business logic program (STOCK_LEDGER_PURGE) will invoke a call to a new program specific for handling historical tables such as HALF_DATA table, etc. that are considered partitioned tables. PARTITION_SQL.PURGE_INTERVAL_PARTITION is

called passing each target table names "HALF_DATA", "DAILY_DATA", "WEEK_DATA", and "MONTH_DATA" This called program will execute the proper deletion/purging of records from target table by exercising table partitioning handling such as Dropping Interval Partition (same as truncate or delete from table).

Scheduling Constraints

Table 19–1 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 12 hours interval - once a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	NA

Restart/Recovery

NA

Locking Strategy

NA

Security Considerations

NA

Performance Considerations

NA

Key Tables Affected

Table 19–2 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
SYSTEM_VARIABLES	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
ALL_PART_TABLES	Yes	No	No	No
ALL_TAB_PARTITIONS	Yes	No	No	No
HALF_DATA	No	No	No	Yes
DAILY_DATA	No	No	No	Yes
WEEK_DATA	No	No	No	Yes
MONTH_DATA	No	No	No	Yes

I/O Specification

NA

saleoh (End Of Half Rollup of Data/Calculations for Stock Ledger)

Module Name	saloeh.pc
Description	End Of Half Rollup of Data/Calculations for Stock Ledger
Functional Area	Stock Ledger
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS337
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The End of Half Stock Ledger Processing is different from many of the other 'End of' processes in that it is also the program that controls how many months of stock ledger data remain on the tables, in addition to the updates to the Half Data table. This program should be run after the end-of-month processing for month 6 has run and before the end-of-month processing for month 1 has run.

The first step for this program is to delete records from stock ledger tables that are 18 months or older. Specifically, the tables that are deleted from are DAILY_DATA, WEEK_DATA, MONTH_DATA, HALF_DATA, and HALF_DATA_BUDGET. The 18-month limit is not a system parameter - it is hard-coded into the program.

The next step in this program is for new records to be written into HALF_DATA_ BUDGET for each department/location for next year's half.

This program also rolls up the inter-stock take shrink amount and inter-stock take sales amount from the HALF_DATA table at the department/location level for this half and calculates the shrinkage percent to insert into HALF_DATA_BUDGET for the next year's half.

Restart/Recovery

There is no main driving cursor for this program. The different functions of this batch program have their own driving cursors. All the driving cursors are threaded by department using the v_restart_dept view. The logical unit of work (LUW) for the delete functions is a half number while the different insert functions have the following LUWs

- half_data() - dept/class/subclass/location
- half_data_budget() - dept/location

Data is committed every time the number of rows processed exceeds commit_max_ctr.

Design Assumptions

N/A

salprg (Purge Stock Ledger History)

Module Name	salprg.pc
Description	Purge Stock Ledger History
Functional Area	Stock Ledger
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS344
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is used to purge old transaction-level stock ledger records from the Transaction Data History table. The Retain Transaction Data (TRAN_DATA_RETAINED_DAYS_NO) system parameter is used to define how many days the Transaction Data History records should be kept in the system. This program will be run nightly to remove any records older than the current date - the "Retain Transaction Data" days.

Restart/Recovery

N/A

Design Assumptions

N/A

stkledgr_hist_purge_job (Purge Stock Ledger History)

Module Name	stkledgr_hist_purge_job
Description	Purge Stock Ledger History
Functional Area	Stock Ledger
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (STKLEDGR_HIST_PURGE_THREAD) will filter eligible records from general ledger key mapping (KEY_MAP_GL) table based on its purge criteria from system parameter settings. The Retain Transaction Data Days (tran_data_retained_days_no) parameter will determine how many days the Transaction Data History records should be kept in the system. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_STKLEDGR_HIST_PURGE_STG.

The Business logic program (STKLEDGR_HIST_PURGE_THREAD) will process all records from the staging table. Using bulk processing, this program will delete the records from general ledger key mapping (KEY_MAP_GL) table. PARTITION_SQL.PURGE_INTERVAL_PARTITION is also called passing the target table name "TRAN_DATA_HISTORY" and will execute the proper deletion/purging of records from target table by exercising table partitioning handling such as Dropping Interval Partition (same as truncate or delete from table). It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 19–3 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 12 hours interval - once a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 19–4 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_STKLEDGR_HIST_PURGE_STG	Yes	Yes	No	Yes

Table 19–4 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
TRAN_DATA_HISTORY	No	No	No	Yes
KEY_MAP_GL	No	No	No	Yes

Design Assumptions

NA

nwppurge (Purge of Aged End of Year Inventory Positions)

Module Name	nwppurge.pc
Description	Purge of Aged End of Year Inventory Positions
Functional Area	Stock Ledger
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS277
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program purges the records from the table NWP after a certain amount of years have passed. The number of years is held in the configurable system level parameter NWP_RETENTION_PERIOD.

Restart/Recovery

Restart/recovery is not applicable, but the records will be committed based on the commit max counter setup in the restart control table.

Design Assumptions

- NWP refers to 'Niederstwertprinzip' and is a legal German accounting financial inventory reporting requirement for calculating year-end inventory position based on the last receipt cost.
- The NWP Indicator system parameter supports this German specific inventory reporting requirement. For German customers, this needs to be 'Y' to allow for the annual NWP calculations & processes.
- This is not relevant for customers outside Germany.

nwp_purge_job (Purge of Aged End of Year Inventory Positions)

Module Name	nwp_purge_job

Description	Purge of Aged End of Year Inventory Positions
Functional Area	Stock Ledger
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (NWP_PURGE_THREAD) will filter eligible records from year-end inventory position (NWP) table based on its purge criteria from system parameter settings. The NWP Retention Period (nwp_retention_period) parameter will determine certain amount of years have passed for NWP records before purging. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_NWP_PURGE_STG.

The Business logic program (NWP_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from year-end inventory position (NWP) table. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 19–5 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad-Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 19–6 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No

Table 19–6 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_NWP_PURGE_STG	Ys	Yes	No	Yes
NWP	Yes	No	No	Yes

Design Assumptions

- NWP refers to 'Niederstwertprinzip' and is a legal German accounting financial inventory reporting requirement for calculating year-end inventory position based on the last receipt cost.
- The NWP Indicator system parameter supports this German specific inventory reporting requirement. For German customers, this needs to be 'Y' to allow for the annual NWP calculations & processes.
- This is not relevant for customers outside Germany.

nwpyearend (End of Year Inventory Position Snapshot)

Module Name	nwpyearend.pc
Description	End of Year Inventory Position Snapshot
Functional Area	Stock Count
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RMS278
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program takes a snapshot of the item's stock position and cost at the end of the year. When the end of year NWP snapshot process runs, it takes a snapshot of stock and weighted average cost (WAC) for every item/location combination currently holding stock. If there is not a record already on the NWP table for an item/location/year combination in the snapshot, a new record is added for that item/location/year combination.

Restart/Recovery

The logical unit of work for this program is set at the location/item level. Threading is done by supplier using the v_restart_store_wh view to thread properly. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O. The changes will be posted when the commit_max_ctr value is reached and the value of the counter is subject to change based on implementation.

Design Assumptions

- NWP refers to 'Niederstwertprinzip' and is a legal German accounting financial inventory reporting requirement for calculating year-end inventory position based on the last receipt cost.
- The NWP Indicator system parameter supports this German specific inventory reporting requirement. For German customers, this needs to be 'Y' to allow for the annual NWP calculations & processes.
- This is not relevant for customers outside Germany.

stlgdnld (Weekly or Historical Download of Stock Ledger Data)

Module Name	stlgdnld.pc
Description	Weekly or Historical Download of Stock Ledger Data
Functional Area	Stock Ledger
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS17
Wrapper Script	batch_stlgdnld.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program extracts stock ledger data at the item level. The program can extract data for a historic period or for the most current complete week. The program accepts an input file that determines whether the extract is a historic extract or a weekly extract.

This program is often used in integration with RPAS applications.

Restart/Recovery

The logical unit of work for this program is set at item, location type, location and date. Threading is done by dept using the v_restart_dept view to thread properly.

The changes will be posted when the commit_max_ctr value is reached. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O. The value of the counter is subject to change based on implementation.

I/O Specification

Integration Type Download from Merchandising

File Name	The input filename is a runtime parameter. The output filename is hardcoded to stkldgr%d.dat where %d is substituted with the domain id. Each run of the program can produce multiple output files, one for each department. Additional input parameters are defined in the input file
Integratin Contract	IntCon000034 (output file)

Input File Layout

Table 19–7 Input File Layout

Field Name	Field Type	Default Value	Description
Task Indicator	Char(1)	N/A	Task Indicator. Valid values are 'H' - historical, 'W' - weekly
From Date	Char(8)	N/A	From Date in 'YYYYMMDD' format
To Date	Char(8)	N/A	To Date in 'YYYYMMDD' format

Output File Layout

Table 19–8 Output File Layout

Field Name	Field Type	Default Value	Description
Item	Char(25)	N/A	Item number
Location Type	Char(1)	N/A	Location Type Valid values are 'S','W'
Location	Number(20)	N/A	Location Number
Eow_date	Char(8)	N/A	End of Week date in 'YYYYMMDD' format
Update_Ind	Char(1)	N/A	Update Indicator Valid values are 'I ' and 'U'
Regular_sales_retail	Number(25,4)	N/A	Regular sales value (retail)
Regular_sales_cost	Number(25,4)	N/A	Regular sales value (cost)
Regular_sales_units	Number(17,4)	N/A	Regular sales value (units)
Promo_sales_retail	Number(25,4)	N/A	Promo sales value (retail)
Promo_sales_cost	Number(25,4)	N/A	Promo sales value (cost)

Table 19–8 (Cont.) Output File Layout

Field Name	Field Type	Default Value	Description
Promo_sales_units	Number(17,4)	N/A	Promo sales value (units)
Clear_sales_retail	Number(25,4)	N/A	Clearance sales value (retail)
Clear_sales_cost	Number(25,4)	N/A	Clearance sales value (cost)
Clear_sales_units	Number(17,4)	N/A	Clearance sales value (units)
Sales_retail_excluding_vat	Number(25,4)	N/A	Sales value excluding vat (retail)
Custom_returns_retail	Number(25,4)	N/A	Custom returns value (retail)
Custom_returns_cost	Number(25,4)	N/A	Custom returns value (cost)
Custom_returns_units	Number(17,4)	N/A	Custom returns value (units)
Rtv_retail	Number(25,4)	N/A	Return to Vendor value (retail)
Rtv_cost	Number(25,4)	N/A	Return to Vendor value (cost)
Rtv_units	Number(17,4)	N/A	Return to Vendor value (units)
Reclass_in_retail	Number(25,4)	N/A	Reclass In value (retail)
Reclass_in_cost	Number(25,4)	N/A	Reclass In value (cost)
Reclass_in_units	Number(17,4)	N/A	Reclass In value (units)
Reclass_out_retail	Number(25,4)	N/A	Reclass Out value (retail)
Reclass_out_cost	Number(25,4)	N/A	Reclass Out value (cost)
Reclass_out_units	Number(17,4)	N/A	Reclass Out value (units)
Perm_markdown_value	Number(25,4)	N/A	Permanent markdown value (retail)
Prom_markdown_value	Number(25,4)	N/A	Promotion markdown value (retail)
Clear_markdown_value	Number(25,4)	N/A	Clearance markdown value (retail)
Markdown_cancel_value	Number(25,4)	N/A	Markdown cancel value
Markup_value	Number(25,4)	N/A	Markup value

Table 19–8 (Cont.) Output File Layout

Field Name	Field Type	Default Value	Description
Markup_cancel_value	Number(25,4)	N/A	Markup cancel value
Stock_adj_retail	Number(25,4)	N/A	Stock adjustment value (retail)
Stock_adj_cost	Number(25,4)	N/A	Stock adjustment value (cost)
Stock_adj_units	Number(17,4)	N/A	Stock adjustment value (units)
Received_retail	Number(25,4)	N/A	Received value (retail)
Received_cost	Number(25,4)	N/A	Received value (cost)
Received_units	Number(17,4)	N/A	Received value (units)
Tsf_in_retail	Number(25,4)	N/A	Transfer In value (retail)
Tsf_in_cost	Number(25,4)	N/A	Transfer In value (cost)
Tsf_in_units	Number(17,4)	N/A	Transfer In value (units)
Tsf_out_retail	Number(25,4)	N/A	Transfer Out value (retail)
Tsf_out_cost	Number(25,4)	N/A	Transfer Out value (cost)
Tsf_out_units	Number(17,4)	N/A	Transfer Out value (units)
Freight_cost	Number(25,4)	N/A	Freight cost
Employee_disc_retail	Number(25,4)	N/A	Employee disc (retail)
Cost_variance	Number(25,4)	N/A	Cost variance
Wkroom_other_cost_sales	Number(25,4)	N/A	Wkroom other sales (cost)
Cash_disc_retail	Number(25,4)	N/A	Cash disc (retail)
Freight_claim_retail	Number(25,4)	N/A	Freight Claim (retail)
Freight_claim_cost	Number(25,4)	N/A	Freight Claim (cost)
Freight_claim_units	Number(25,4)	N/A	Freight Claim (Units)
Stock_adj_cogs_retail	Number(25,4)	N/A	Stock Adjust COGS (retail)
Stock_adj_cogs_cost	Number(25,4)	N/A	Stock Adjust COGS (cost)
Stock_adj_cogs_units	Number(25,4)	N/A	Stock Adjust COGS (Units)

Table 19–8 (Cont.) Output File Layout

Field Name	Field Type	Default Value	Description
Intercompany_in_retail	Number(25,4)	N/A	Intercompany In value (retail)
Intercompany_in_cost	Number(25,4)	N/A	Intercompany In value (cost)
Intercompany_in_units	Number(25,4)	N/A	Intercompany In value (units)
Intercompany_out_retail	Number(25,4)	N/A	Intercompany Out value (retail)
Intercompany_out_cost	Number(25,4)	N/A	Intercompany Out value (cost)
Intercompany_out_units	Number(25,4)	N/A	Intercompany Out value (units)
Intercompany_markup	Number(25,4)	N/A	Intercompany Markup
Intercompany_markup_ units	Number(25,4)	N/A	Intercompany Markup (units)
Intercompany_markdown	Number(25,4)	N/A	Intercompany Markdown
Intercompany_markdown_ units	Number(25,4)	N/A	Intercompany Markdown (units)
Wo_activity_upd_inv	Number(25,4)	N/A	Work Order Activity - Update Inventory (cost)
Wo_activity_upd_inv_units	Number(25,4)	N/A	Work Order Activity - Update Inventory (units)
Wo_activity_post_fin	Number(25,4)	N/A	Work Order Activity - Post to Financials (retail)
Wo_activity_post_fin_units	Number(25,4)	N/A	Work Order Activity - Post to Financials (units)

Design Assumptions

N/A

otbdlsal (Open To Buy Download Stock Ledger)

Module Name	otbdlsal.pc
Description	Open To Buy Download Stock Ledger
Functional Area	OTB - Stock Ledger to Planning System Interface
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS16

Wrapper Script Rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module will sum stock ledger data from the DAILY_DATA table and opening stock information from the WEEK_DATA table across the current week, grouping by department, class, subclass, location and date, and export the data to a flat file for use by an outside planning system.

Restart/Recovery

The logical unit of work for the OTBDLSAL module is department, class, subclass and location. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of the file I/O. The recommended commit counter setting is 10000 records. Each time the record counter equals the maximum recommended commit number, an application image array record will be written to the restart_start_array for restart/recovery if a fatal error occurs.

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	OTB - Stock Ledger to Planning System Interface IntCon00030

Output File Format

Table 19–9 File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Sequence Number	Number(10)	0000000001	Keeps track of the record's position in the file by line number
	File Type Definition	Char(4)	STKE	Identifies file as Stock Ledger Export
	File Create Date	Char(14)	vdate	Date file was written by batch program in YYYYMMDD format. Remaining six characters are blank.
FDETL	File Type Record Descriptor	Char(5)	FDETL	Identifies file record type
	File Line Sequence Number	Number(10)	line number in file	Keeps track of the record's position in the file by line number
	Transaction Set Control Number	Number(14)	sequence number	Used to force unique file check
	Department	Number(4)	N/A	The ID number of a department
	Class	Number(4)	N/A	The ID number of a class within the department given
	Subclass	Number(4)	N/A	The ID number of a subclass within the class given

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Loc_type	Char(1)	N/A	The type of the location from which stock ledger data was collected
	Location	Number(10)	N/A	The location from which stock ledger data was collected
	Half No.	Number(5)	N/A	The half number for this stock ledger data
	Month No.	Number(2)	N/A	The month number in the half for this stock ledger data
	Week No.	Number(2)	N/A	The week number in the month for this stock ledger data
	Open Stock Retail	Number(20,4)	N/A	The retail opening stock from the week_data table *10000 (implied 4 decimal places) for this stock ledger period
	Open Stock Cost	Number(20,4)	N/A	The cost opening stock from the week_data table *10000 (implied 4 decimal places) for this stock ledger period
	Stock Adjustments Retail	Number(20,4)	N/A	The retail stock adjustments summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Stock Adjustments Cost	Number(20,4)	N/A	The cost stock adjustments summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Purchases Retail	Number(20,4)	N/A	The retail purchases summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Purchases Cost	Number(20,4)	N/A	The cost purchases summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	RTV Retail	Number(20,4)	N/A	The retail return to vendor amount summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	RTV Cost	Number(20,4)	N/A	The cost return to vendor amount summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Freight Cost	Number(20,4)	N/A	The freight cost summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Net Sales Retail	Number(20,4)	N/A	The retail net sales summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Net Sales Cost	Number(20,4)	N/A	The cost net sales summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Returns Retail	Number(20,4)	N/A	The retail returns amount summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Returns Cost	Number(20,4)	N/A	The cost returns amount summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Promotional Markdowns Retail	Number(20,4)	N/A	The retail promotional markdowns summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Markdown Cancellations Retail	Number(20,4)	N/A	The retail markdown cancellations summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Employee Discount Retail	Number(20,4)	N/A	The retail employee discounts amount summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Workroom Amount	Number(20,4)	N/A	The workroom amount summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Cash Discount Amount	Number(20,4)	N/A	The cash discounts amount summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Sales Units	Number(12,4)	N/A	The sales units summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Markups Retail	Number(20,4)	N/A	The retail markups summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Markup Cancellations Retail	Number(20,4)	N/A	The retail markup cancellations summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Clearance Markdowns Retail	Number(20,4)	N/A	The retail clearance markdowns summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Permanent Markdowns Retail	Number(20,4)	N/A	The retail permanent markdowns summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Freight Claim Retail	Number(20,4)	N/A	The retail freight claim summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Freight Claim Cost	Number(20,4)	N/A	The cost freight claim summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Stock Adjust Cost of Goods Sold (COGS) Retail	Number(20,4)	N/A	The retail stock adjust COGS summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Stock Adjust Cost of Goods Sold (COGS) Cost	Number(20,4)	N/A	The cost stock adjust COGS summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Inter-company In Retail	Number(20,4)	N/A	The Inter-company In retail summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Inter-company In Cost	Number(20,4)	N/A	The Inter-company In cost summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Inter-company Out Retail	Number(20,4)	N/A	The Inter-company Out Retail summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Inter-company Out Cost	Number(20,4)	N/A	The Inter-company Out Cost summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Inter-company Markup	Number(20,4)	N/A	The Inter-company Markup summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Inter-company Markdown	Number(20,4)	N/A	The Inter-company Markdown summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
	Work Order Activity Update Inventory	Number(20,4)	N/A	The Work Order Activity Update Inventory summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period

Table 19–9 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Work Order Activity Post Finishing	Number(20,4)	N/A	The Work Order Activity Post Finishing summed from the DAILY_DATA table *10000 (implied 4 decimal places) for this stock ledger period
FTAIL	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Sequence Number	Number(10)	N/A	Keeps track of the record's position in the file by line number
	Control Number File Line Count	Number(10)	N/A	Total number of all transaction lines, not including file header and trailer

Design Assumptions

N/A

trandataload.ksh (External Transaction Data Upload)

Module Name	trandataload.ksh
Description	External Transaction Data Upload
Functional Area	Finance
Module Type	Integration
Module Technology	KSH
Catalog ID	RMS 376
Wrapper Script	batch_trandataload.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This process, along with trandataprocess.ksh, provides a mechanism to write records directly into the TRAN_DATA tables based on a file from an external system. The

primary purpose of this functionality is to allow additional costs to be included in stock ledger valuation that cannot be included based on existing Merchandise functionality. Records written to the TRAN_DATA tables do not necessarily have a connection to any Merchandising transaction, and are based on a determination made outside of Merchandising. The records written through this mechanism function exactly the same as records written by normal Merchandising processes. For cost based transactions, the information must be passed at an item/location level. For retail-based transactions, it can be at either an item/location or subclass/location level.

Note: There is no support for recalculating or impacting unit inventory in Merchandising based on the transactions passed in, and only cost or retail value in the stock ledger is impacted - although the weighted average cost (WAC) may also be impacted if that method of accounting is used in Merchandising

The trandataload script loads the staging table STAGE_EXT_TRAN_DATA table from a flat file using SQL Loader and divides the data into chunks to be processed in parallel threads based on the commit_max_counter and num_threads value on RESTART_CONTROL table.

This script accepts the following input parameters:

- Database Connect string
- File load indicator – This indicator is passed as Y if a flat file has to be loaded into the table STAGE_EXT_TRAN_DATA else its N
- Input file – This is the path of the input file. This is mandatory when File load indicator is Y.

The SQL loading from a flat file is optional in the script. If File load indicator is Y the program validates if the input file exists and logs an error in case the input file does not exist. The SQL Load (sqlldr) process loads the input file using control file - trandataload.ctl into the STAGE_EXT_TRAN_DATA table.

- A fatal error from sqlldr will halt the process.
- Rejected records are a non-fatal error and loader will continue processing and create bad file and discard files in case the input file does not match the expected format.

If you chose not to load data into the staging table (File load indicator 'N') then the batch assumes that data has been loaded on the staging table from a different source. After the loading process is complete, the batch divides the data into chunks. If the staging table is empty or all the records are in 'P'rocessed status then the batch logs an appropriate error.

Chunking Logic

- Dense rank the staged records over Subclass, item and location.
- Divide the rank value by the commit max counter.
- Rounding the divided value gives the Chunk ID to which the particular value belongs to.
- Item can be NULL on the staging table, when NULL consider item to be '-999'.
- This will make sure the records with same subclass value and having item as NULL and NOT NULL are not grouped together in a chunk.

Since records with item have to be processed differently, (WAC recalculation and Variance postings) the batch makes sure that they fall in a different chunk to those records which do not have item value.

The Chunk data is inserted into STAGE_EXT_TRAN_DATA_CHUNK table.

Restart/Recovery

N/A

I/O Specification - Input File Specification

This batch uses SQL Loader to populate the staging table. The input file should be in pipe delimited format. Sample record structure would look like:

```
<item>|<dept>l<class>|<subclass>|<location>|<loc_type>|<tran_date>|<tran_
code>|<adj_code>|<units>|<total_cost>|<total_retail>|<ref_no_1>|<ref_no_2>|<GL_
ref_no>|<Old_unit_retail>|<New_unit_retail>|<Sales_type>|<VAT_rate>|<av_
cost>|<ref_pack_no>|<total_cost_excl_elc>|<WAC_reclculate_ind>|<status>|<create_
timestamp>|
```

File Layout

The table below specifies the detail of each field in the record.

Table 19–10 File Layout

Field Name	Field Type	Default Value	Description
Item	VARCHAR2(25)	N/A	Item is an optional field. Transactions can be uploaded at the Subclass level also.
Dept	NUMBER(4)	N/A	Mandatory Field
Class	NUMBER(4)	N/A	Mandatory Field
Subclass	NUMBER(4)	N/A	Mandatory Field
Location	NUMBER(10)	N/A	Mandatory Field
Loc_type	VARCHAR2(1)	N/A	Valid values - 'S', 'W', 'E'
Tran_data	DATE	N/A	Mandatory Field
Tran_code	NUMBER(2)	N/A	Mandatory Field
Adj_code	VARCHAR2(1)	N/A	Valid values - 'C', 'U', 'A'
Units	NUMBER(12, 4)	N/A	Mandatory Field
Total_cost	NUMBER(20, 4)	N/A	N/A
Total_retail	NUMBER(20, 4)	N/A	N/A
Ref_no_1	NUMBER(10)	N/A	N/A
Ref_no_2	NUMBER(10)	N/A	N/A
Gl_ref_no	NUMBER(10)	N/A	N/A
Old_unit_retail	NUMBER(20, 4)	N/A	N/A
New_unit_retail	NUMBER(20, 4)	N/A	N/A

Table 19–10 (Cont.) File Layout

Field Name	Field Type	Default Value	Description
Pgm_name	VARCHAR(100)	N/A	N/A
Sales_type	VARCHAR2(1)	N/A	Valid values - 'C', 'R', 'P'
Vat_rate	NUMBER(12, 4)	N/A	N/A
Av_cost	NUMBER(20, 4)	N/A	N/A
Ref_pack_no	VARCHAR2(25)	N/A	N/A
Total_cost_excl_elc	NUMBER(20, 4)	N/A	N/A
Wac_recalculate_ind	VARCHAR2(1)	N/A	If Weighted Average Cost of the Item-Location should be recalculated after uploading this transaction then this value should be passed as 'Y'.
Status	VARCHAR2(1)	'N'	This value will be defaulted to 'N' by this program. It will be updated to 'P' once it has been processed else to 'E' in case of Error.
Create_timestamp	DATE	Sysdate	N/A

Design Assumptions

N/A

trandataprocess.ksh (External Transaction Data Process)

Module Name	trandataprocess.ksh
Description	External Transaction Data Process
Functional Area	Finance
Module Type	Business Processing
Module Technology	KSH
Catalog ID	RMS377
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This process, along with trandataload.ksh, provides a mechanism to write records directly into the TRAN_DATA tables based on a file from an external system. The

primary purpose of this functionality is to allow additional costs to be included in stock ledger valuation that cannot be included based on existing Merchandise functionality. Records written to the TRAN_DATA tables do not necessarily have a connection to any Merchandising transaction, and are based on a determination made outside of Merchandising. The records written through this mechanism function exactly the same as records written by normal Merchandising processes. For cost based transactions, the information must be passed at an item/location level. For retail-based transactions, it can be at either an item/location or subclass/location level.

Note: There is no support for recalculating or impacting unit inventory in Merchandising based on the transactions passed in, and only cost or retail value in the stock ledger is impacted - although the weighted average cost (WAC) may also be impacted if that method of accounting is used in Merchandising.

Trandataprocess batch processes the data on STAGE_EXT_TRAN_DATA and inserts into the TRAN_DATA table. This batch should be run after trandataload.ksh.

This batch validates the records on the staging table. The status records that fail validation are updated to 'E'rror on the staging table with error message.

The records which pass the validations are inserted into TRAN_DATA table and Weighted Average Cost is recalculated in case the WAC_recalc_ind is 'Y' for the record.

This script accepts the following input parameters:

- Database Connect string.
- Number of parallel threads - optional parameter. This is to override the value set on RESTART_CONTROL table.

This script calls the TRAN_DATA_IMPORT_SQL to import the transaction records on STAGE_EXT_TRAN_DATA table that haven't been processed yet. Each thread of the program processes a single chunk of data. After processing the Chunk, the status of the chunk is updated to 'P'rocessed.

The batch program performs the below validations on the staged records before inserting to TRAN_DATA. Status of the records which fail validations will be updated to 'E'rror on STAGE_EXT_TRAN_DATA along with the reasons for validation failure.

- Validates Dept, Class, and Subclass against SUBCLASS table.
- Validates location and loc_type against STORE and WH tables.
- Validates tran_code against TRAN_DATA_CODES table.
- If Item is not NULL validate if the item exists and is a transaction level item.
- If Item is not NULL validate if the item belongs to the dept/class/subclass.
- If Item not NULL validate if it is ranged to the location.
- Validate that item is not a pack.
- Item can be NULL only if it belongs to a Retail accounting department.
- When RECAL_WAC_IND = 'Y', ITEM and TOTAL_COST should not be NULL.
- Both total_cost and total_retail cannot be null.
- The loc_type should be 'W' or 'S' or 'E'.
- For TRAN_CODES - 37, 38, 63 and 64, GL_REF_NO should not be NULL

- For TRAN_CODES - 22 and 23 total cost should not be NULL
- For TRAN_CODES - 11, 12, 13, 14, 15, 16, 60, 80, and 81, total retail should not be NULL or total cost should be NULL.
- For TRAN_CODES - 1, 4, 20, 24, 27, 30, 31, 37 and 38, total cost should not be NULL OR (total_retail should not be NULL and sellable_ind is 'Y')

Once records are validated, the batch program calculates the Weighted Average Cost (WAC) for the records with WAC_RECALC_IND = 'Y'. In case the calculated WAC <= 0 and if there is inventory present the location then a cost variance record (TRAN_CODE - 70) is inserted into TRAN_DATA. Cost variance transaction is also posted for those item locations which have no or negative inventory.

Restart/Recovery

N/A

Design Assumptions

N/A

20

Franchise Management

To scale up business operations and market presence, particularly in new markets, retailers may choose to utilize business partners to manage branded or co-branded stores while retaining the retailer's business processes and value proposition. Businesses who partner with a retailer to expand the retailer's presence are known as franchisees. Franchisees may operate one or more stores under the retailer's banner. Merchandising supports two types of franchise management:

1. Franchise inventory is managed by the retailer

   For this scenario, the retailer owns/manages the retail experience through planning, ordering, selling and tracking of inventory at franchise stores. In Merchandising, it is assumed that franchise customer locations will be set up as stockholding stores, with a store type of "Franchise".

2. Franchise inventory is not managed by the retailer

   For this case, the retailer does not own or manage inventory, but mandatorily requires a franchise customer to adhere to business processes across franchise stores. This may also include retailers with smaller scale wholesale operations constitute a small fraction of the retailers business. For both these scenarios, it is assumed that non-stockholding stores will be setup in Merchandising to represent these franchise customer locations.

The batch processes that are used for Franchise Management in Merchandising fall primarily into the following areas:

Customers

Merchandising maintains customer groups and customers pertaining to franchise operations as a hierarchy above customer locations. Customer groups and customers can be entered in Merchandising or uploaded from an external system. Customer locations are set up as franchise stores in Merchandising and can be designated as either stockholding or non-stockholding.

Costing

For all items that are 'sold' to franchise customer locations from a retailer, a selling price must be determined. The default selling price for franchise stores is calculated and held on the future cost table as the pricing cost. To calculate the cost, Merchandising uses the concept of templates and it is a template's association with a franchise store and merchandise hierarchy that determines the value on the future cost table. Cost templates and their relationships with franchise locations/merchandise hierarchies can be entered into Merchandising or uploaded via a batch process.

Franchise Orders

Franchise orders need to be raised in order to fulfill demand from a franchise customer. A franchise order is considered a sales order between the retailer and the franchise customer. A franchise order contains the item requisition to be sourced from a certain location (vendor, company warehouse or store) and fulfilled at one or more franchise stores by one or more required need dates. A franchise order also contains the price at which the items on the order will be sold to the franchise customer. Franchise Orders can be entered into Merchandising via one of the following methods:

1. Manually via the Franchise Sales Order screen.
2. From an external application using the WF Order Upload (wfordupld) batch.
3. Automatically through replenishment, store orders, item requests, AIP generated POs/Transfers and Allocations for stockholding franchise stores.

Once a franchise order is created and approved, a transfer (for warehouse or store sourced orders) or purchase order (for supplier sourced orders) will be created to manage the inventory movement. All franchise orders must be for a single customer.

Franchise Returns

Franchise returns are used whenever inventory moves from a franchise store back to a company owned location. Franchise returns cannot be created directly back to a supplier, it is assumed they will always first come back to a company owned location. Unlike franchise orders, which can be created for multiple franchise stores, franchise returns are always from a single franchise store. A franchise return contains the items being returned and the return price. If known, the original franchise order is referenced with the return and the price from the original order is used as a default. Like franchise orders, franchise returns can be created in three different ways:

1. Manually via the Franchise Returns screen.
2. From an external application using the WF Return Upload (wfretupld) batch.
3. Automatically through store-initiated transfers or transfers sent from an external system for stockholding franchise stores.

Batch Design Summary

The following batch designs are included in this functional area:

- fcosttmplupld.ksh (Upload Cost Buildup Template)
- fcosttmplprocess.ksh (Process Cost Buildup Template Upload)
- fcosttmplpurge.ksh (Purge Staged Cost Template Data)
- wf_cost_template_purge_job (Purge Staged Cost Template Data)
- fcustomerupload.ksh (Franchise Customer Upload)
- fcustomerprocess.ksh (Process Uploaded Franchise Customers and Customer Groups)
- fcustupldpurge.ksh (Franchise Customer Staging Purge)
- wfordupld.ksh (Franchise Order Upload)
- wf_apply_supp_cc.ksh (Apply Supplier Cost Change to Franchise Orders)
- wfordcls.pc (Franchise Order Close)

- wf_orders_close_job (Franchise Order Close)
- wfordprg.pc (Franchise Order Purge)
- wf_orders_purge_job (Franchise Order Purge)
- wfretupld.ksh (Franchise Return Upload)
- wfretcls.pc (Franchise Return Close)
- wf_returns_close_job (Franchise Return Close)
- wfrtnprg.pc (Franchise Return Purge)
- wf_returns_purge_job (Franchise Return Purge)
- wfslsupld.ksh (Upload of Franchise Sales to RMS)
- wfbillex.ksh (Franchise Billing Extract)

fcosttmplupld (Upload Cost Buildup Template)

Module Name	fcosttmplupld.ksh
Description	Upload Cost Buildup Template
Functional Area	Franchise Management
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS125
Wrapper Script	rmswrap_shell_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module uploads cost buildup templates and franchise cost relationships used for franchise pricing from an external system into Merchandising staging tables. It also performs both technical and business validation of the data sent in the file; for example, it validates that start and end dates are included for new and updated templates.

Note: No date format is specified in the input file, as any valid PL/SQL date format can be used.

Restart/Recovery

The restart recovery is different from the conventional Merchandising batch. There are three points on the batch upload process where users can evaluate the successful load of the data.

- SQL load - SQL load dumps invalid records that do not meet certain technical requirements (for example:. file layout issues, data type inconsistencies, and so on.). The rejected record is written either to a bad file or to a discard file. The

discard file contains records that do not satisfy conditions such as missing or invalid record types. Records with other technical issues are written to the bad file.

Note: A non-fatal code is returned by the program and a message will be written to the log file if reject files are created

---

Action Required: When such conditions exist, you may update either the bad or discard file and attempt to reload using the same files.

1. Business Validation Level - the data from the files are loaded into the staging tables for validation. PL/SQL functions determine if this loaded data is valid enough to be inserted into the actual Merchandising tables. Records that do not meet certain technical or business validations are rejected and the information is updated back into the staging table with an appropriate error message and the batch issues a NON-FATAL return code.

   Action Required: When this condition exists, you can fix the data upload file and try to reload.

2. Chunking validated data - At this point the data from staging tables that have passed business validation are chunked based on the number of valid transactions (cost templates) and max_chunk_size from RMS_PLSQL_BATCH_CONFIG table. If there are no valid transactions to be chunked, batch issues a FATAL return code.

   Action Required: When this condition exists, you can fix the data upload file and try to reload.

Key Tables Affected

Table 20–1 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_WF_COST_TMPL_UPLD_FHEAD	Yes	Yes	Yes	No
SVC_WF_COST_TMPL_UPLD_THEAD	Yes	Yes	Yes	No
SVC_WF_COST_TMPL_UPLD_TDETL	Yes	Yes	Yes	No
SVC_WF_COST_TMPL_UPLD_TTAIL	Yes	Yes	Yes	No
SVC_WF_COST_TMPL_UPLD_FTAIL	Yes	Yes	Yes	No
SVC_WF_COST_TMPL_UPLD_STATUS	Yes	Yes	Yes	Yes
ELC_COMP	Yes	No	No	No
STORE	Yes	No	No	No
CLASS	Yes	No	No	No
SUBCLASS	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
RMS_PLSQL_BATCH_CONFIG	Yes	No	No	No

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000021

SQL Loader Input File Layout

Table 20–2 SQUL Loader Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char(5)	N/A	Identifies file record type. Valid value is FHEAD.
	File Line Identifier	Number(10)	N/A	Sequential file line number
	File Type Definition	Char(5)	CTMPL	Identifies file as 'Cost Template Upload'
	File Create Date	Date	SYSDATE	Date on which the file was created by external system
Transaction Header	File Record Descriptor	Char(5)	N/A	Identifies transaction header record type. Valid value is THEAD
	File Line Identifier	Number(10)	N/A	Sequential file line number

Table 20–2 (Cont.) SQUL Loader Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Message Type	Char(30)	N/A	Identifies the action that will be performed on the franchise cost template header information that is provided as part of this record It can be either create or update or delete a franchise cost template. Valid message types are: costtmpadd (for additions), costtmpmod (for updates), costtmpdel (for deletions)
	Template ID	Number(10)	N/A	Template ID
	Template Description	Char(120)	N/A	Template Description
	Template Type	Char(1)	N/A	Indicates the type of the template. Valid values are M = Margin then Up-Charge, U = Up-charges, then Margin, R = % of Retail and C = Cost
	Percentage	Number(12,4)	N/A	Margin percent or % off Retail value; required if template type is M, U and R types of templates

Table 20–2 (Cont.) SQUL Loader Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Cost	Number(20,4)	N/A	Indicates the franchise cost for an item when template type is 'C' This is mandatory and should only be populated if template type is 'C'
	Final Cost	Char(1)	N/A	Signifies if the cost is final or acquisition. Valid values are 'Y' or 'N'
Transaction Detail	File Record Descriptor	Char(5)		Identifies transaction detail record type. Valid value is TDETL
	File Line Identifier	Number(10)		Sequential file line number
	Message Type	Char(30)		Identifies the action that will be performed on the franchise cost template relationship information that is provided as part of this record. It can be either create or update or delete a cost relationship. Valid values are: costtmpreladd (for additions), costtmprelmod (for updates), costtmpreldel (for deletions)
	Dept	Number(4)		Department associated with the cost template

Table 20–2 (Cont.) SQUL Loader Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Class	Number(4)		Class associated with the cost template
	Subclass	Number(4)		Subclass associated with the cost template
	Item	Char(25)		Unique number that identifies a valid item associated with the template. Used for template types of 'C' only
	Location	Number(10)		Franchise Store Number associated with the template
	Start Date	Date		Date on which a cost template will be effective for the subclass/item and franchise store (required for update and delete of a cost relationship)
	End Date	Date		Date on which a cost template will expire for a subclass/item and franchise store (required for update and delete of a cost relationship)
	New Start Date	Date		New Date on which a franchise cost relationship will be effective
	New End Date	Date		New Date on which a franchise cost relationship will expire

Table 20–2 (Cont.) SQUL Loader Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Cost Component ID	Char(10)		Unique code which signifies the up-charge cost component when First_ Applied is 'U' This should only be populated if First Applied is 'U'
Transaction Trailer	File Record Descriptor	Char(5)	N/A	Identifies transaction trailer record type. Valid value is TTAIL
	File Line Identifier	Number(10)	N/A	Sequential file line number
	Transaction Record Counter	Number(10)	N/A	Number of TDETL records in this transaction set
File Trailer	File Record Descriptor	Char(5)	N/A	Identifies file trailer record type. Valid value is TTAIL
	File Line Identifier	Number(10)	N/A	Sequential file line number
	File Record Counter	Number(10)	N/A	Number of records/transactions processed in current file (only records between FHEAD & FTAIL)

Design Assumptions

N/A

fcosttmplprocess (Process Cost Buildup Template Upload)

Module Name	fcosttmplprocess.ksh
Description	Process Cost Buildup Template Upload
Functional Area	Franchise Management
Module Type	Business Processing
Module Technology	ksh

Catalog ID RMS224

Wrapper Script batch_fprocess.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module processes franchise cost buildup templates and franchise cost relationships that were uploaded from an external source into staging tables and loads them from the staging tables into Merchandising base tables. The module is designed to process inserts, updates and deletes for these data elements.

Restart/Recovery

The restart recovery is different from the conventional Merchandising batch.

During the batch process, you can evaluate the successful processing of data in the following way:

PL/SQL function will load the data from staging tables into Merchandising tables. For records that result (insert/update/delete) in constraint error or are not found in the Merchandising tables (for update/delete) are rejected and the information is updated back in the corresponding staging table with appropriate error message. Also, records that do not meet certain business validations (which can only be validated during data processing) are rejected and the information is updated back in the corresponding staging table with appropriate error message.

Action Required: When this condition exists, you can fix the data upload file and try to reload and process the data.

Key Tables Affected

Table 20–3 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_WF_COST_TMPL_UPLD_FHEAD	Yes	No	Yes	No
SVC_WF_COST_TMPL_UPLD_THEAD	Yes	No	Yes	No
SVC_WF_COST_TMPL_UPLD_TDETL	Yes	No	Yes	No
SVC_WF_COST_TMPL_UPLD_TTAIL	Yes	No	Yes	No
SVC_WF_COST_TMPL_UPLD_FTAIL	Yes	No	Yes	No
SVC_WF_COST_TMPL_UPLD_STATUS	Yes	No	Yes	No
WF_COST_BUILDUP_TMPL_HEAD	Yes	Yes	Yes	Yes
WF_COST_BUILDUP_TMPL_DETAIL	Yes	Yes	Yes	Yes

Table 20–3 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
WF_COST_RELATIONSHIP	Yes	Yes	Yes	Yes
GTT_WF_COST_ RELATIONSHIP	No	Yes	No	No
COST_EVENT_COST_ RELATIONSHIP	No	Yes	No	No
COST_EVENT	No	Yes	No	No
COST_EVENT_RESULT	No	Yes	No	No
COST_EVENT_THREAD	No	Yes	No	Yes
FUTURE_COST_GTT	No	Yes	No	No
FUTURE_COST	No	No	No	Yes

Design Assumptions

N/A

fcosttmplpurge (Purge Staged Cost Template Data)

Module Name	fcosttmplpurge.ksh
Description	Purge Staged Cost Template Data
Functional Area	Franchise Management
Module Type	Admin
Module Technology	ksh
Catalog ID	RMS225
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module purges data from the staging tables used by the Cost Buildup Template Upload process. The module is designed to purge all the data from the staging tables that have passed the system parameter Foundation Staging Retention days.

Restart/Recovery

N/A

Key Tables Affected

Table 20–4 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_WF_COST_TMPL_UPLD_FHEAD	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_THEAD	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_TDETL	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_TTAIL	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_FTAIL	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_STATUS	No	No	No	Yes
SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

N/A

wf_cost_template_purge_job (Purge Staged Cost Template Data)

Module Name wf_cost_template_purge_job

Description Purge Staged Cost Template Data

Functional Area Franchise Management

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of one step processing only. It will retain the business logic processing from original KSH script algorithm.

The Business logic program (CORESVC_WF_COST_TMPL_UPLD_SQL.PURGE_RECORDS) will removed all old/aged records from the staging tables used by the Cost Buildup Template Upload process which have passed the purge criteria from the system parameter setting, Foundation Staging Retention Days (fdn_stg_retention_days).

Scheduling Constraints

Table 20–5 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc

Table 20–5 (Cont.) Scheduling Constraints

Schedule Information	Description
Frequency	Daily - 1 hour interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	NA

Restart/Recovery

NA

Key Tables Affected

Table 20–6 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_WF_COST_TMPL_UPLD_FHEAD	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_THEAD	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_TDETL	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_TTAIL	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_FTAIL	No	No	No	Yes
SVC_WF_COST_TMPL_UPLD_STATUS	No	No	No	Yes
SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

NA

fcustomerupload (Franchise Customer Upload)

Module Name	fcustomerupload.ksh
Description	Franchise Customers Upload
Functional Area	Franchise Management
Module Type	Integration
Module Technology	ksh
Integration Catalog ID	RMS126
Wrapper Script	rmswrap_shell_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module uploads franchise customers and customer group details from an external system into Merchandising staging tables. It also performs both technical and business validation of the data sent in the file; for example, it validates that a customer cannot be deleted if a franchise store is associated with it.

Restart/Recovery

The restart recovery is different from the conventional Merchandising batch. There are three points on the batch upload process where you can evaluate the successful load of the data.

- SQL load - SQL load dumps invalid records that do not meet certain technical requirements (for example:. data type inconsistencies, and so on.). The rejected record is written either to a bad file or to a discard file. The discard file contains records that do not satisfy conditions such as missing or invalid record types. Records with other technical issues are written to the bad file.

  Note: A non-fatal code is returned by the program and a message will be written to the log file if reject files are created.

  Action Required: When such conditions exist, you may update either the bad or discard file and attempt to reload using the same files.

- File-Based Validations - the data from the files are loaded into the staging tables for validation. PL/SQL functions will validate the data in the staging tables to determine if there are any issues with the FHEAD and FTAIL in the file. These kinds of errors are FATAL errors and the batch ends the file processing immediately with return code 255.

  Action Required: When this condition exists, you can fix the data upload file and try to reload.

- Business Validation Level - PL/SQL functions determine if the transactions loaded are valid enough to modify the actual Merchandising tables. Records that do not meet certain technical or business validations are rejected and the information is updated back into the staging table with an appropriate error message and the batch issues a NON-FATAL return code 1.

  Action Required: When this condition exists, you can fix the data upload file and try to reload.

Key Tables Affected

Table 20–7 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_FCUSTUPLD_FHEAD	Yes	Yes	Yes	No
SVC_FCUSTUPLD_THEAD	Yes	Yes	Yes	No
SVC_FCUSTUPLD_TDETL	Yes	Yes	Yes	No

Table 20–7 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_FCUSTUPLD_TTAIL	Yes	Yes	Yes	No
SVC_FCUSTUPLD_FTAIL	Yes	Yes	Yes	No
SVC_FCUSTUPLD_STATUS	Yes	Yes	Yes	No
WF_CUSTOMER_GROUP	Yes	No	No	No
WF_CUSTOMER	Yes	No	No	No
STORE	Yes	No	No	No

I/O Specification

Integration Type Upload to Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000022

Input File Layout

Table 20–8 File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Record Descriptor	Char(5)	N/A	Identifies file record type. It should be FHEAD
	File Line ID	Number(10)	N/A	ID of current line being processed by input file
	File Type	Char(5)	FCUST	Identifies file as 'Franchise customer upload'
	File Create Date	Date	SYSDATE	Date file was written by external system

Table 20–8 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
Transaction Header	File Record Descriptor	Char(5)	N/A	Identifies transaction record type. It should be THEAD
	File Line ID	Number(10)	N/A	ID of current line being processed by input file
	Message Type	Char(30)	N/A	Identifies the action that will be performed on the franchise customer transaction header record. It can be either create (fcustgrpcre) or update (fcustgrpupd) or delete (fcustgrpdel) a franchise customer group
	Franchise Customer group ID	Number(10)	N/A	Customer group ID
	Franchise Customer group Name	Char(120)	N/A	Customer group name. This field is optional for delete

Table 20–8 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
Transaction Detail	File Record Descriptor	Char(5)	N/A	Identifies transaction record type. It should be TDETL
	File Line ID	Number(10)	N/A	ID of current line being processed by input file
	Message Type	Char(30)	N/A	Identifies the action that will be performed on the franchise customer transaction detail record. It can be either create (fcustcre) or update (fcustupd) or delete (fcustdel) a franchise customer .
	Franchise Customer ID	Number(10)	N/A	Customer ID to be processed
	Franchise Customer Name	Char(120)	N/A	Customer Name
	Credit Ind	Char(1)	N	This field will determine if the franchise customer has good credit. Valid values are Y and N
	Auto approve Ind	Char(1)	N	To auto approve the externally uploaded orders and returns. Valid values are Y and N

Table 20–8 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
Transaction Trailer	File Record Descriptor	Char(5)	N/A	Identifies file record type. It should be TTAIL
	File Line ID	Number(10)	N/A	ID of current line being processed by input file
	Transaction Record Count	Number(10)	N/A	Number of TDETL records in this transaction set.(total records between THEAD & TTAIL)
File Trailer	File Record Descriptor	Char(5)	N/A	Identifies file record type. It should be FTAIL
	File Line ID	Number(10)	N/A	ID of current line being processed by input file.
	File Record Counter	Number(10)	N/A	Number of records/transactions processed in current file (total records between FHEAD & FTAIL)

Design Assumptions

N/A

fcustomerprocess (Process Uploaded Franchise Customers and Customer Groups)

Module Name	fcustomerprocess.ksh
Description	Process Uploaded Franchise Customers and Customer Groups
Functional Area	Franchise Management
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS492

Wrapper Script batch_fprocess.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module processes the franchise customer groups and franchise customers information from the staging tables and loads it into Merchandising base tables for franchise customer groups and franchise customer information. It is also designed to process (insert/update or delete) the validated data that maps to franchise customer groups and franchise customer information.

Restart/Recovery

The restart recovery is different from the conventional Merchandising batch. During the batch process, you can evaluate the successful processing of data in the following way:

- PL/SQL function will load the data from staging tables into Merchandising tables. For records that result (insert/update/delete) in constraint error or are not found in the Merchandising tables(for update/delete) are rejected and the information is updated back in the corresponding staging table with appropriate error message.

  Also, records that do not meet certain business validations (which can only be validated during data processing) are rejected and the information is updated back in the corresponding staging table with appropriate error message.

Action Required: When this condition exists, you can fix the data upload file and try to reload and process the data.

Commit Points

Commit points are performed per transaction.

Key Tables Affected

Table 20–9 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_FCUSTUPLD_FHEAD	Yes	No	Yes	No
SVC_FCUSTUPLD_THEAD	Yes	No	Yes	No
SVC_FCUSTUPLD_TDETL	Yes	No	Yes	No
SVC_FCUSTUPLD_TTAIL	Yes	No	Yes	No
SVC_FCUSTUPLD_FTAIL	Yes	No	Yes	No
SVC_FCUSTUPLD_STATUS	Yes	No	Yes	No
WF_CUSTOMER_GROUP	Yes	Yes	Yes	Yes
WF_CUSTOMER	Yes	Yes	Yes	Yes
STORE	Yes	No	No	No

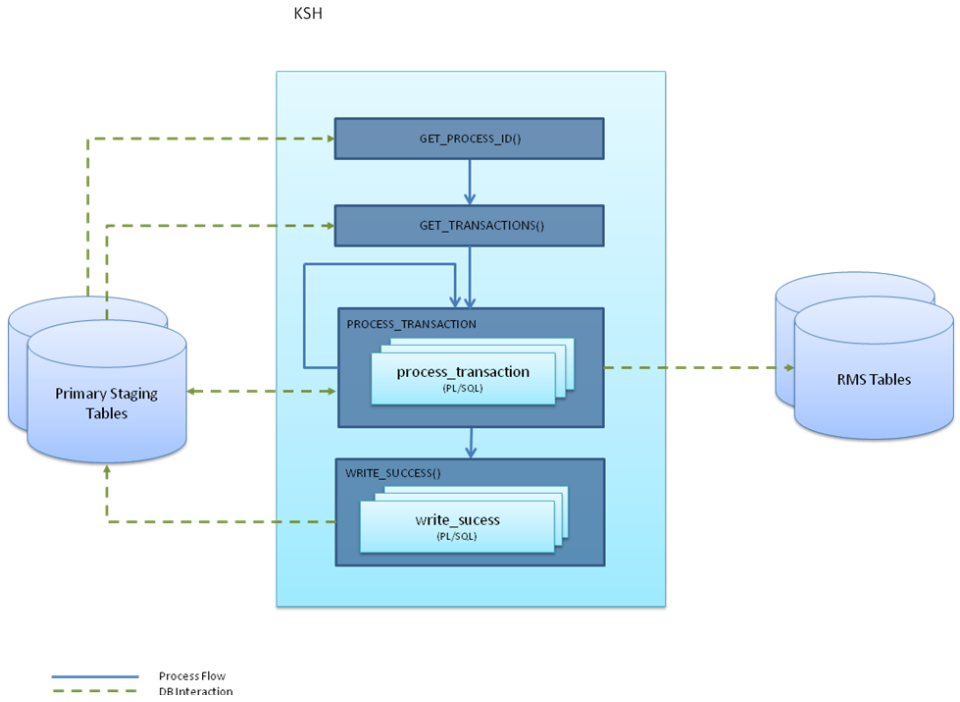
Design Assumptions

N/A

Program Flow

This diagram describes the process flow of the fcustomerprocess.ksh module.

Figure 20–1 Process Flow



fcustupldpurge (Franchise Customer Staging Purge)

Module Name	fcustomerupldpurge.ksh
Description	Franchise Customer Staging Purge
Functional Area	Franchise Management
Module Type	Admin
Module Technology	ksh
Catalog ID	RMS493
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module purges data from the staging tables used by the Franchise Customer Upload and Franchise Customer Process scripts. It is designed to purge all the data

from the staging tables that have passed the system parameter for Foundation Staging Retention days.

Restart/Recovery

N/A

Key Tables Affected

Table 20–10 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_FCUSTUPLD_FHEAD	No	No	No	Yes
SVC_ FCUSTUPLD_ THEAD	No	No	No	Yes
SVC_ FCUSTUPLD_TDETL	No	No	No	Yes
SVC_ FCUSTUPLD_TTAIL	No	No	No	Yes
SVC_ FCUSTUPLD_FTAIL	No	No	No	Yes
SVC_ FCUSTUPLD_ STATUS	No	No	No	Yes
SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

N/A

wfordupld.ksh (Franchise Order Upload)

Module Name	wfordupld.ksh
Description	Franchise Order Upload
Functional Area	Franchise Management
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS60
Wrapper Script	batch_wfupload.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to upload franchisee orders from an external source. These orders will be created with an order type of 'EDI' and will be created for the source type specified in the upload file. If source type is not specified, then the costing location for the item/franchise store will be used. Orders will be created in approved status if the customer is setup for auto approval, assuming that the customer has valid credit.

If the customer fails credit check or if available inventory at the source location is insufficient to fulfill the order, the order will be generated in input status.

Franchise orders from customers that are not identified for 'Auto Approval' are uploaded into Merchandising in input status. These orders will need to be manually approved in Merchandising in order to be considered active.

Restart/Recovery

The restart recovery is different from the conventional Merchandising batch. There are two points on the batch upload process where users can evaluate the successful load of the data.

- SQL load - At this point, SQL load dumps invalid records that do not meet certain technical requirements (for example:. file layout issues, data type inconsistencies, and so on.). The rejected record is written to a bad file or to a discard file. The discard file contains records that do not satisfy conditions, such as missing or invalid record types. Records with other technical issues are written to the bad file.

  Note: A non-fatal code is returned by the program and a message will be written to the log file if reject files are created.

  Action Required: When such conditions exist, you may update either the bad or discard file and attempt to reload using the same files.

- Business Validation - At this point data from the file(s) are loaded into the staging table(s). PL/SQL functions determine if this loaded data is valid enough to be inserted into the actual Merchandising tables. For records that do not meet certain technical or business validations, the error message will be updated in staging table.

  Action Required: When this condition exists, you can fix the data upload file and try to reload the file with valid data.

Key Tables Affected

Table 20–11 Key Tables Affected

Table	Select	Insert	Update	Delete
FUTURE_COST	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPPLIER	Yes	No	No	No
REPL_ITEM_LOC	Yes	No	No	No
STORE_ORDERS	Yes	No	No	No
SVC_WF_ORD_HEAD	Yes	Yes	Yes	No
SVC_WF_ORD_DETAIL	Yes	Yes	Yes	No
SVC_WF_ORD_TAIL	Yes	Yes	Yes	No
SYSTEM_OPTIONS	Yes	No	No	No

Table 20–11 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
WF_COST_RELATIONSHIP	Yes	No	No	No
WF_COST_BUILDUP_ TMPL_HEAD	Yes	No	No	No
WF_CUSTOMER	Yes	No	No	No
WF_ORDER_HEAD	Yes	Yes	No	No
WF_ORDER_DETAIL	Yes	Yes	No	No
WF_ORDER_EXP	No	Yes	No	No

I/O Specification

Integration Type Download from Merchandising

File Name wford*.dat

Integration Contract IntCon000108

SQL Loader Input File Layout

Table 20–12 SQL Loder Input File Layout

Record Name	Field Name	Field Type	Null allowed?	Default Value	Description
FHEAD	File head descriptor	Char(5)	No	FHEAD	Describes file line type.
	Line Number	Number(10)	No	N/A	Id of the current line being processed.
	Customer Id	Number(10)	No	N/A	Customer ID of the customer requesting the order.
	Customer Order Reference number	Char(20)	No	N/A	A reference field used by the customer for their tracking purposes.
	Currency Code	Char(3)	No	N/A	This is the currency on which the order was transacted.
	Default Billing location	Number(10)	No	N/A	A customer's location where the billing for the entire order is sent. If blank, each location is billed.
	Comments	Char(2000)	Yes	N/A	Any other miscellaneous information relating to the order.
FDETL	File record descriptor	Char(5)	No	FDETL	Describes file line type.
	Line Number	Number(10)	No	N/A	Id of the current line being processed.
	Item	Char(25)	No	N/A	The item on the franchise order.
	Customer Location	Number(10)	No	N/A	The franchise store requesting the order.

Table 20–12 (Cont.) SQL Loder Input File Layout

Record Name	Field Name	Field Type	Null allowed?	Default Value	Description
	Source Loc Type	Char(2)	Yes	N/A	Source location type for which the franchise order has been created. Valid values are ST - Store, WH - warehouse, or SU - Supplier
	Source Location	Number(10)	Yes	N/A	Source location for which the franchise order has been created.
	Requested Quantity	Number (12,4)	No	N/A	Number of item units being ordered, includes 4 implied decimal places
	Unit of Purchase	Char(3)	No	N/A	Unit of purchase can be the item's standard unit of measure, case, inners or pallets.
	Fixed Cost	Number (20,4)	Yes	N/A	This is cost which will be charged to the customer for the item on the franchise order; value includes 4 implied decimal places.
	Need Date	Char(11)	No	N/A	Date on which the item is needed in the franchise store, with the following format "DD-MON-YYYY' .

Table 20–12 (Cont.) SQL Loder Input File Layout

Record Name	Field Name	Field Type	Null allowed?	Default Value	Description
	Not After Date	Char(11)	No	N/A	Date after which the item may no longer be accepted for a franchise store, with the following format "DD-MON-YY YY'.
FTAIL	File record descriptor	Char(5)		FTAIL	Marks end of file.
	Line Number	Number(10)		N/A	Id of current line being processed.
	File record count	Number(10)		N/A	Number of detail records.

Design Assumptions

N/A

wf_apply_supp_cc.ksh (Apply Supplier Cost Change to Franchise Orders)

Module Name	wf_apply_supp_cc.ksh
Description	Apply Supplier Cost Change to Franchise Orders
Functional Area	Franchise Management
Module Type	Business Processing
Module Technology	ksh
Catalog ID	RMS389
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This function updates approved franchise orders for supplier sourced records whose items/franchise stores are impacted by supplier cost changes. Only those item/franchise store combinations that use cost templates based on supplier cost or have not had a fixed cost defined on the order are eligible to be updated. Only those supplier cost changes that were flagged as recalculating orders result in this update.

Restart/Recovery

N/A

Key Tables Affected

Table 20–13 Key Tables Affected

Table	Select	Insert	Update	Delete
WF_ORDER_HEAD	Yes	No	No	No
WF_ORDER_DETAIL	No	No	Yes	No
WF_ORDER_EXP	No	Yes	No	Yes
FUTURE_COST	Yes	No	No	No
COST_SUSP_SUP_HEAD	Yes	No	No	No
COST_SUSP_SUP_DETAIL	Yes	No	No	No
COST_SUSP_SUP_DETAIL_LOC	Yes	No	No	No
WF_COST_RELATIONSHIP	Yes	No	No	No
WF_COST_BUILDUP_TMPL_HEAD	Yes	No	No	No
MV_CURRENCY_CONVERSION_RATES	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No

Design Assumptions

- The pricing cost for franchise orders in input or pending credit approval status is not updated because the order cost will be updated based on any changes on franchise order approval.

wfordcls (Franchise Order Close)

Module Name	wfordcls.pc
Description	Franchise Order Close
Functional Area	Franchise Management
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS391
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to close the Franchise orders if the conditions below are met:

- Franchise Order is not in Input (I) or Requires Credit Approval (R) status.
- All the transfers associated with the franchise order are in closed/deleted status.

- All the allocations associated with franchise order are in closed status.
- All the purchase orders associated with franchise order are in closed status.
- Store orders associated with franchise order do not have a null processed date or a need qty > 0.

Restart/Recovery

The logical unit of work for this module is defined as a unique franchise order number. The restart franchise order view is used for threading. This batch program uses table-based restart/recovery. The commit happens in the database when the maximum commit counter is reached.

Key Tables Affected

Table 20–14 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
WF_ORDER_HEAD	Yes	No	Yes	No
TSFHEAD	Yes	No	No	No
STORE_ORDERS	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	No

Design Assumptions

N/A

wf_orders_close_job (Franchise Order Close)

Module Name	wf_orders_close_job
Description	Franchise Order Close
Functional Area	Franchise Management
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (WF_ORDERS_CLOSE_THREAD) will filter eligible records from franchise order header (WF_ORDER_HEAD) table based on its conditions are met:

- Franchise Order is not in Input (I) or Requires Credit Approval (R) status.

- All the transfers associated with the franchise order are in closed/deleted status.
- All the allocations associated with franchise order are in closed status.
- All the purchase orders associated with franchise order are in closed status.
- Store orders associated with franchise order do not have a null processed date or a need qty > 0.

These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_WF_ORDERS_CLOSE_STG.

The Business logic program (WF_ORDERS_CLOSE) will process all records from the staging table. Using bulk processing, this program will update the records from franchise order header (WF_ORDER_HEAD) table to "D" (Closed) status. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 20–15 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 1 hour interval - 12 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by franchise order number

Restart/Recovery

NA

Key Tables Affected

Table 20–16 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_WF_ORDERS_ CLOSE_STG	Yes	Yes	No	Yes
WF_ORDER_HEAD	Yes	No	Yes	No
TSFHEAD	Yes	No	No	No
STORE_ORDERS	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No

Table 20–16 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ALLOC_HEADER	Yes	No	No	No

Design Assumptions

NA

wfordprg (Franchise Order Purge)

Module Name	wfordprg.pc
Description	Franchise Order Purge
Functional Area	Franchise Management
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS392
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to purge franchise orders from Merchandising after a set number of days have elapsed, as defined by the system parameter Franchise History Months. Additionally, in order to be purged via this process, the franchise orders must have no associated franchise returns and must not have any billing records that have not been extracted or where not enough time has elapsed since they were extracted, as defined by the Franchise History Months system parameter.

Restart/Recovery

The logical unit of work for this module is defined as a unique franchise order number. The restart franchise order view is used for threading. This batch program uses table-based restart/recovery. The commit happens in the database when the maximum commit counter is reached.

Key Tables Affected

Table 20–17 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
WF_ORDER_HEAD	Yes	No	No	Yes
WF_ORDER_DETAIL	Yes	No	No	Yes

Table 20–17 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
WF_BILLING_SALES	Yes	No	No	Yes
WF_ORDER_AUDIT	No	No	No	Yes
WF_ORDER_EXP	No	No	No	Yes
TSFHEAD	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No
STORE_ORDERS	Yes	No	No	No

Design Assumptions

- Transfers, Allocations, POs and Store Orders associated with franchise orders are deleted through purge processes for those functional areas (e.g. tsfprg for Transfers). Franchise orders will not be allowed to be deleted until these associated records have been removed via the other processes.

wf_orders_purge_job (Franchise Order Purge)

Module Name wf_orders_purge_job

Description Franchise Order Purge

Functional Area Franchise Management

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (WF_ORDERS_PURGE_THREAD) will filter eligible records from franchise orders header (WF_ORDER_HEAD) table based on its conditions are met and number of days elapsed as defined by system parameter setting, Franchise History Months:

- All Franchise Order Details have its NOT_AFTER_DATE not yet elapsed as declared by the system parameter setting.
- All the franchise returns associated with the franchise order were deleted or purged.
- All the billing records associated with the franchise order are not yet extracted or where not enough time has elapsed since they were extracted as defined by the system parameter setting.
- All transfers, Orders and Store Orders associated with the franchise order were purged through their respective purge process.

These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_WF_ORDERS_PURGE_STG.

The Business logic program (WF_ORDERS_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from franchise orders header (WF_ORDER_HEAD) and other related franchise order tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 20–18 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by WF Order number

Restart/Recovery

NA

Key Tables Affected

Table 20–19 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_WF_ORDERS_ PURGE_STG	Yes	Yes	No	Yes
WF_ORDER_HEAD	Yes	No	No	Yes
WF_ORDER_DETAIL	Yes	No	No	Yes
WF_BILLING_SALES	Yes	No	No	Yes
WF_ORDER_AUDIT	No	No	No	Yes
WF_ORDER_EXP	No	No	No	Yes
TSFHEAD	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No
STORE_ORDERS	Yes	No	No	No

Design Assumptions

NA

wfretupld.ksh (Franchise Return Upload)

Module Name	wfretupld.ksh
Description	Franchise Return Upload
Functional Area	Franchise Management
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS154
Wrapper Script	batch_wfupload.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used for uploading franchise returns sent from an external source, such as an external order management application. When returns are uploaded in this manner, the data will be validated and the return will be created in Merchandising. Additionally, an associated franchise return transfer will also be created.

Restart/Recovery

The restart recovery is different from the conventional Merchandising batch. There are two points on the batch upload process where users can evaluate the successful load of the data.

- SQL load - At this point, SQL load dumps invalid records that do not meet certain technical requirements (for example:. file layout issues, data type inconsistencies, and so on.). The rejected record is written either to a bad file or to a discard file. The discard file contains records that do not satisfy conditions, such as missing or invalid record types. Records with other technical issues are written to the bad file.

  Note: A non-fatal code is returned by the program and a message will be written to the log file if reject files are created. When such conditions exist, you may either update the bad or discard file and attempt to reload using the same files.

- Business Validation - At this point data from the file(s) are loaded into the staging table(s). PL/SQL functions determine if this loaded data is valid enough to be inserted into the actual Merchandising tables. For all records that do not meet certain technical or business validations, the error message will be updated in staging table. When this condition exists, you can fix the data upload file and try to reload the file with valid data.

Key Tables Affected

Table 20–20 Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_WF_RET_HEAD	Yes	Yes	Yes	No
SVC_WF_RET_DETAIL	Yes	Yes	Yes	No
SVC_WF_RET_TAIL	Yes	Yes	Yes	No
WF_RETURN_HEAD	Yes	Yes	No	No
WF_RETURN_DETAIL	Yes	Yes	No	No
TSFHEAD	Yes	Yes	Yes	No
TSFDETAIL	Yes	Yes	No	No
ITEM_LOC	Yes	Yes	No	No
ITEM_LOC_SOH	Yes	Yes	Yes	No
TRAN_DATA	Yes	Yes	No	No

I/O Specification

Integration Type Upload to Merchandising

File Name wfreturn*.dat

Integration Contract Intcon000109

SQL Loader Input File Layout

The following is the file pattern for the upload file.

Note: The values are pipe "|" delimited and can optionally be enclosed by " ".

Table 20–21 SQL Loader Input File Layout

Record Name	Field Name	Field Type	Null Allowed?	Default Value	Description
FHEAD	File head descriptor	Char(5)	No	FHEAD	Describes file line type.
	Line Number	Number(10)	No		Id of the current line being processed.
	Customer ID	Number(10)	No		Franchise customer ID of the customer making the return.
	Customer Return Reference number	Char(20)	No		A reference field used by the franchise customer for their tracking purposes.
	Currency Code	Char(3)	No		This is the return currency.
	Comments	Char(2000)	Yes		Any other miscellaneous information related to the return.
FDETL	File record descriptor	Char(5)	No	FDETL	Describes file line type.
	Line Number	Number(10)	No	N/A	Id of the current line being processed.
	Item	Char(25)	No	N/A	The item on the franchise return.
	Franchise Order Number	Number(10)	No	N/A	The franchise order number against which the return is made.
	Customer Location	Number(10)	No	N/A	The franchise location which is making the return.

Table 20–21 (Cont.) SQL Loader Input File Layout

Record Name	Field Name	Field Type	Null Allowed?	Default Value	Description
	Return Loc Type	Char(1)	No	N/A	Return location type for the franchise return; valid values are S - store or W - warehouse.
	Return Location	Number(10)	No	N/A	Return location for the franchise return.
	Return Method	Char(1)	No	N/A	The type of return; valid values are: -R-Return to Store/Warehouse -D-Destroy at site
	Unit of measure	Char(3)	No	N/A	The unit measure of the return quantity. This is assumed to be the items standard UOM.
	Return qty	Number(12,4)	No	N/A	The quantity of item to be returned
	Return Reason	Char(6)	No	N/A	Return reason code; valid values are found on the CODE_DETAIL table where CODE_TYPE is 'RTVR'.
	Return unit cost	Number(20,4)	Yes	N/A	The per unit cost for the return.
	Restock Type	Char(1)	No	N/A	Indicates how the restocking fee will be calculated per item; valid values are S-specific or V-value.
	Restock Fee	Number(20,4)	No	N/A	Unit restocking fee.

Table 20–21 (Cont.) SQL Loader Input File Layout

Record Name	Field Name	Field Type	Null Allowed?	Default Value	Description
FTAIL	File record descriptor	Char(5)	No	FTAIL	Marks end of file.
	Line Number	Number(10)	No	N/A	Id of current line being processed.
	File record count	Number(10)	No	N/A	Number of detail records.

Design Assumptions

N/A

wfretcls (Franchise Return Close)

Module Name	wfretcls.pc
Description	Franchise Return Close
Functional Area	Franchise Management
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS394
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to close franchise returns that are not in input status where all the associated transfers for the return are either in closed or deleted status.

Restart/Recovery

The logical unit of work for this module is defined as a unique return order number. The restart franchise return view is used for threading. This batch program uses table-based restart/recovery. The commit happens in the database when the maximum commit counter is reached.

Key Tables Affected

Table 20–22 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
WF_RETURN_HEAD	Yes	No	Yes	No

Table 20–22 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
TSFHEAD	Yes	No	No	No

Design Assumptions

N/A

wf_returns_close_job (Franchise Return Close)

Module Name	wf_returns_close_job
Description	Franchise Return Close
Functional Area	Franchise Management
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (WF_RETURNSS_CLOSE_THREAD) will filter eligible records from franchise return header (WF_RETURN_HEAD) table based on its conditions are met:

- Franchise Returns is not in Input (I) or Closed (D) status.
- All the transfers associated with the franchise return are in closed/deleted status.

These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_WF_RETURNS_CLOSE_STG.

The Business logic program (WF_RETURNS_CLOSE) will process all records from the staging table. Using bulk processing, this program will update the records from franchise return header (WF_RETURN_HEAD) table to "D" (Closed) status. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 20–23 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 1 hour interval - 12 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA

Table 20–23 (Cont.) Scheduling Constraints

Schedule Information	Description
Threading Scheme	Threaded by WF Return number

Restart/Recovery

NA

Key Tables Affected

Table 20–24 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_WF_RETURNS_CLOSE_STG	Yes	Yes	No	Yes
WF_RETURN_HEAD	Yes	No	Yes	No
TSFHEAD	Yes	No	No	No

Design Assumptions

NA

wfrtnprg (Franchise Return Purge)

Module Name	wfrtnprg.pc
Description	Franchise Return Purge
Functional Area	Franchise Management
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS396
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to purge franchise returns from the Merchandising after a set number of days have elapsed, as defined by the system parameter Franchise History Months. Additionally, in order to be purged via this process, the franchise returns must have no associated billing records that have not been extracted or where

not enough time has elapsed since they were extracted, as defined by the Franchise History Months system parameter.

Restart/Recovery

The logical unit of work for this module is defined as a unique return order no. The restart franchise return view is used for threading. This batch program uses table-based restart/recovery. The commit happens in the database when the maximum commit counter is reached.

Key Tables Affected

Table 20–25 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
WF_RETURN_HEAD	Yes	No	No	Yes
WF_RETURN_DETAIL	No	No	No	Yes
WF_BILLING_RETURNS	Yes	No	No	Yes
TSFHEAD	Yes	No	No	No

Design Assumptions

- Transfers associated with franchise returns are deleted through the Transfer Purge (tsfprg) process. Franchise returns will not be allowed to be deleted until these associated records have been removed via that process.

wf_returns_purge_job (Franchise Return Purge)

Module Name wf_returns_purge_job

Description Franchise Return Purge

Functional Area Franchise Management

Module Type Admin - Ad hoc

Module Technology Background Processing

Catalog ID NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (WF_RETURNS_PURGE_THREAD) will filter eligible records from franchise returns header (WF_RETURN_HEAD) table based on its conditions are met and number of days elapsed as defined by system parameter setting, Franchise History Months:

- All the billing records associated with the franchise order are not yet extracted or where not enough time has elapsed since they were extracted as defined by the system parameter setting.

- All transfers associated with the franchise order were purged through their respective purge process.

These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_WF_RETURNS_PURGE_STG.

The Business logic program (WF_RETURNS_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from franchise returns header (WF_RETURN_HEAD) and other related franchise return tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 20–26 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad hoc
Frequency	Daily - 3 hours interval - 4 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by WF Return number

Restart/Recovery

NA

Key Tables Affected

Table 20–27 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_WF_RETURNS_ PURGE_STG	Yes	Yes	No	Yes
WF_RETURN_HEAD	Yes	No	No	Yes
WF_RETURN_DETAIL	No	No	No	Yes
WF_BILLING_RETURNS	Yes	No	No	Yes
TSFHEAD	Yes	No	No	No

Design Assumptions

NA

wfslsupld.ksh (Upload of Franchise Sales to Merchandising)

Module Name	wfslsupld.ksh
Description	Upload of Franchise Sales to Merchandising
Functional Area	Franchise Management
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS156
Wrapper Script	batch_wfslsupld.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Non-stockholding franchise stores in Merchandising are used for retailers who have franchise or other business customers for whom they supply inventory, but don't manage it for them. However, even though inventory information will not be available for these locations in Merchandising, sales information will be able to be uploaded to Merchandising via this process to allow retailers to have better visibility to future demand from these customers. In addition to uploading sales information, this same batch script also purges old non-stockholding franchise store sales records from Merchandising. The script runs in 4 modes:

- Load - this mode will load the data from the file into a staging table in Merchandising for processing; any errors encountered in validating the data on the upload are also written to the staging table (WFSLSUPLD_STAGING).
- Process - this mode will process the records in the staging table that did not have errors during load, which includes both writing the data to the WF_NONSTOCKHOLDING_SALES table, as well as purging the processed records from the staging table.
- Reject - this mode will process the records on the staging table that had errors on initial load. It will create a reject file for each location/report date with the data in error for that location/date. The records will then be deleted from the staging table.
- Purge - this mode is used to purge old sales records from the WF_NON_STOCKHOLDING_SALES table. Records are deleted based on the system parameter Non-stockholding Franchise Sales History days (WF_NON_STOCK_SALES_HIST_DAYS).

Restart/Recovery

The program can be restarted by running the wfslsupld REJECT mode to create an input file of rejected records and wfslsupld LOAD/PROCESS mode to reprocess the rejected records.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Input file name is a parameter during runtime
Integration Contract	IntCon000111

Input File Layout

Table 20–28 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Identifies the file record type
	File Line Id	Char(10)	N/A	Sequential file line number
	File type definition	Char(4)	WFSU	Identifies the file type
	Customer Location	Number(10)	N/A	Store number identifier for the customer location
	Report Date	Char(14)	N/A	Report date of the file in YYYYMMDD HHMMSS format
	File Create Date	Char(14)	N/A	File Create Date in YYYYMMDD HHMMSS format

Table 20–28 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FDETL	Record descriptor	Char(5)	FDETL	Identifies the file record type
	File Line Id	Char(10)	N/A	Sequential file line number
	Item	Char(25)	N/A	Item number identifier
	Net Sales Quantity	Number(12)	N/A	Sales Quantity with 4 implied decimal places
	Net Sales Quantity UOM	Char(4)	N/A	Unit of Measure for the Net Sales Quantity
	Total Retail Amount	Number(20)	N/A	Total Retail Amount with 4 implied decimal places
	Total Retail Amount Currency	Char(3)	N/A	Currency code for the Total Retail Amount
FTAIL	Record descriptor	Char(5)	FTAIL	Identifies the file record type
	File Line Id	Number(10)	N/A	Sequential file line number
	File Record counter	Number(10)	N/A	Number of records/transactions processed in current file (only records between head & tail)

Design Assumptions

N/A

wfbillex.ksh (Franchise Billing Extract)

Module Name	wfbillex.ksh
Description	Franchise Billing Extract
Functional Area	Franchise Management
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS155

Wrapper Script rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this shell script module is to fetch all billing information for Franchise sale and return transactions and write these to an output file for integration with an external financial application that manages billing. A file is generated for each customer location (store)/day.

Restart/Recovery

The logical unit of work for this module is defined as the customer location (store). Only one commit will be done for a customer location that has been completely processed. The WFBX formatted output file will be created with a temporary name and renamed just before a customer location commit. In case of failure, all work done will be rolled back.

I/O Specification

Integration Type Download from Merchandising

File Name WFBX_<store>_<SYSDATE>

Integration Contract IntCon000110

Output File Layout

Table 20–29 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Identifies the file record type
	File Line Id	Char(10)	N/A	Sequential file line number
	File type definition	Char(4)	WFBX	Identifies the file type
	File Create Date	Char(14)	N/A	File Create Date in YYYYMMDD HHMMSS format

Table 20–29 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
THEAD	Record descriptor	Char(5)	THEAD	Identifies the file record type
	File Line Id	Char(10)	N/A	Sequential file line number
	Customer Location	Number(10)	N/A	Franchise store number
	Customer Order Reference Number	Char(20)	N/A	Reference number provided by the franchise customer
	Franchise Order Number	Number(10)	N/A	Franchise Order Number
	Transaction Type	Char(6)	N/A	SALES or RETURN
	RMA Number	Number(10)	N/A	Return Merchandise Authorization Number for the return
	Order Return Date	Number(8)	N/A	Order return date for Return transaction type or Order date for Sale transaction type in YYYYMMDD format
	Shipment Date	Number(8)	N/A	Date on which the item was shipped to the franchise location or returned to the retailer

Table 20–29 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TDETL	Record descriptor	Char(5)	TDETL	Identifies the file record type
	File Line Id	Char(10)		Sequential file line number
	Item	Char(25)		Item sequence number
	Department	Number(4)		Department number of the item
	Class	Number(4)		Class number of the item
	Subclass	Char(4)		Subclass number of the item
	Order Return Quantity	Number(12)		Return quantity with 4 implied decimal places
	Order Return Quantity UOM	Char(4)		Return quantity unit of measure
	Order Return Cost	Number(20)		Return cost for Return transaction type or Customer cost for Sale transaction type. For both it is the per-unit cost
	Freight Cost	Number(20)		Freight associated to the franchise order
	Return Restocking Fee	Number(20)		Unit restocking fee charged for received items
	VAT Code	Char(6)		VAT code for the item
	VAT Rate	Number(20)		VAT rate associated to the VAT code for the item
	Other Order Charges	Number(20)		Other charges for the item

Table 20–29 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TAIL	Record descriptor	Char(5)	TTAIL	Identifies the file record type
	File Line Id	Char(10)	N/A	Sequential file line number
	Tran Record Counter	Number(6)	N/A	Number of TDETL records in this transaction set
FTAIL	Record descriptor	Char(5)	FTAIL	Identifies the file record type
	File Line Id	Number(10)	N/A	Sequential file line number
	File Record counter	Number(10)	N/A	Number of records/transactions processed in current file (only records between head & tail)

Design Assumptions

N/A

21

Competitive Pricing

The Merchandising competitive pricing functionality extracts a competitor's price for an item. Merchandising masters competitor price information. Pricing uses this information to determine if a price review should be performed.

The batch programs in this chapter only need to be run if the retailer uses competitive shopping to track prices at other retailers and wishes to use this information to drive pricing decisions in Pricing.

Batch Design Summary

The following batch designs are included in this functional area:

- cmpupld.pc (Upload Competitor's Prices)
- cmpprg.pc (Purge Aged Competitive Pricing Data)
- comp_pricing_purge_job (Purge Aged Competitive Pricing Data)

cmpupld (Upload Competitor's Prices)

Module Name	cmpupld.pc
Description	Upload Competitor's Prices
Functional Area	Competitive Pricing
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS61
Wrapper Script	rmswrap_in_rej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program is used to upload and process competitor item prices from an external source. The flat file being uploaded can contain pricing data for a completed shopping list or data for a new list of items to be shopped. The module processes data for both features.

Restart/Recovery

This is a file based upload, and file based restart/recovery logic is applied. The commit_max_ctr field should be set to prevent excessive rollback space usage and to reduce the overhead of file I/O. The recommended commit counter setting is 10000 records (subject to change based on experimentation).

Key Tables Affected

Table 21–1 Key Tables Affected

Table	Select	Insert	Update	Delete
COMP_SHOP_LIST	Yes	Yes	No	No
ITEM_MASTER	Yes	No	No	No
CODE_DETAIL	Yes	No	No	No

I/O Specification

Integration Type Upload to Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000007

Input File Layout

Table 21–2 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	CHAR (5)	FHEAD	Value that identifies the record type.
	File Line Identifier	NUMBER (10)	0000000001	Sequential file line number.
	File Type Definition	CHAR(4)	CMPU	Value that identifies the file as that for this program.
	File Create Date	CHAR (14)	N/A	Date when the file was written by the external system. It should be in the YYYYMMDD HH24MISS format.

Table 21–2 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
File Detail	File Type Record Descriptor	CHAR (5)	FDETL	Value that identifies the record type.
	File Line Identifier	NUMBER (10)		Sequential file line number.
	Shopper ID	NUMBER (4)		Numeric value that uniquely identifies the shopper to which the competitive shopping list is assigned.
	Shop Date	CHAR (14)		Date when the competitive shop was performed. It should be in the YYYYMMDD HH24MISS format.
	Item	CHAR (25)		Alphanumeric value that uniquely identifies the transaction level or below transaction level item that was competitively shopped.

Table 21–2 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Competitor ID	NUMBER(10)		Numeric value that uniquely identifies a competitor.
	Competitor Store ID	NUMBER(10)		Numeric value that uniquely identifies a competitor's store.
	Recorded Date	CHAR (14)		Date when the item's retail price was recorded at the competitor's store. It should be in the YYYYMMDD24MISS format.
	Competitive Retail Price	NUMBER(20,4)		Numeric value that represents the retail price at the competitor's store. Format for this value should include four implied decimal places.
	Competitive Retail Type	CHAR(6)	R, P, C	Value that represents the retail type ('R' is for regular; 'P', promotional; and 'C', clearance) that was recorded.
	Promotion Start Date	CHAR (14)		Effective start date of the competitor's price. It should be in the YYYYMMDD HH24MISS format.
	Promotion End Date	CHAR (14)		Effective end date of the competitor's price. It should be in the YYYYMMDD HH24MISS format.

Table 21–2 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Offer Type Code	CHAR(6)		Alphanumeric value that corresponds to a valid offer type (such as,. Coupon, Bonus Card, Pre-priced). Valid values are defined on CODE_ DETAIL table with CODE_ TYPE 'OFTP'.
	Multi-Units	NUMBER(12 ,4)		Numeric value that represents the number of units that must be purchased to qualify for a multi-unit price. An example of a multi-unit price would be 2 for $3.00. There are four implied decimal places.
	Multi-Units Retail	NUMBER(20 ,4)		Numeric value that represents the price for a multi-unit item that was competitively shopped. There should be four implied decimal places.
File Trailer	File Type Record Descriptor	CHAR(5)	FTAIL	Value that identifies the record type.
	File Line Identifier	NUMBER (10)	N/A	Sequential file line number.
	File Record Counter	NUMBER (10)	N/A	Numeric value that represents the number of FDETL records in the file.

Design Assumptions

- Items included in the file must be defined as transaction level items in Merchandising.

cmpprg.pc (Purge Aged Competitive Pricing Data)

Module Name	cmpprg.pc
Description	Purge Aged Competitive Pricing Data
Functional Area	Competitive Pricing
Module Type	Admin
Module Technology	ProC
Catalog ID	RMS198
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program deletes from the competitive price history table and the competitive shopping list table based purge criteria based on system parameter settings. The Competitive Pricing Months parameter will determine how many months competitive price history should be maintained before deletion. The Competitive Pricing List Days parameter will determine how long a requested shopping list should remain on the shopping list table if it is not complete by the requested shop date.

Restart/Recovery

N/A

Key Tables Affected

Table 21–3 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
COMP_PRICE_HIST	Yes	No	No	Yes
COMP_SHOP_LIST	No	No	No	Yes

Design Assumptions

N/A

comp_pricing_purge_job (Purge Aged Competitive Pricing Data)

Module Name	comp_pricing_purge_job
Description	Purge Aged Competitive Pricing Data
Functional Area	Competitive Pricing
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (COMP_PRICING_PURGE_THREAD) will filter eligible records from competitive price history (COMP_PRICE_HIST) and competitive shipping list (COMP_SHIP_LIST) tables based on its purge criteria from system parameter settings. The Competitive Pricing List Days (comp_list_days) parameter will determine how long a requested shopping list should remain on the shopping list table if it is not complete by the requested shop date. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_COMP_PRICING_PURGE_STG.

The Business logic program (COMP_PRICING_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from competitive price history (COMP_PRICE_HIST) and competitive shipping list (COMP_SHIP_LIST) tables. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 21–4 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 6 hours interval - 2 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table 21–5 Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
PURGE_CONFIG_ OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_COMP_PRICING_ PURGE_STG	Yes	Yes	No	Yes
COMP_PRICE_HIST	Yes	No	No	Yes
COMP_SHOP_LIST	Yes	No	No	Yes

Design Assumptions

NA

22

Item Induction

Item induction is a process for importing item related information into Merchandising from an external source. For many retailers, item creation is initiated in a system outside Merchandising. Some retailers receive item information from their vendors, others initiate items in a planning application, and still others use a product lifecycle management (PLM) application, or a product hub (such as, a PIM application).

Merchandising offers a flexible method of importing items, which supports inducting items into Merchandising with a bare minimum of data and provides a working area for enrichment of those items prior to upload into the production tables in Merchandising. Item induction functionality allows users and systems to upload item data into a staging area or directly into Merchandising using any of the below modes

- Batch
- RIB
- Manual upload

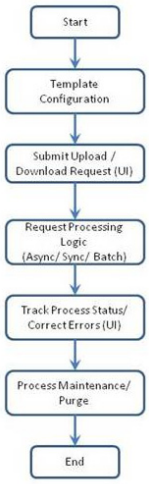
Data uploaded into the staging area though any of the above modes can be downloaded into a spreadsheet, enriched and re-uploaded into the staging area or into Merchandising. Maintainence of items that already existing in Merchandising can also be achieved by downloading the data into a spreadsheet which in turn offers mass maintenance, filtering, and sorting capabilities.

The processing of upload or download requests of item data through manual and batch options is linked to a template definition that specifies which tables and columns are to be made available to you or system for data entry and update. Templates can be created based on user role, business line, item type, and so on. and provide the flexibility to define default values for one or more fields.

Overall management of data in the staging area is achieved through provision of a dedicated purge batch.

For more information on the RIB options for uploading items into the staging area, see the *Oracle Retail Merchandising Foundation Cloud Service Operations Guide, Volume 2 - Message Publication and Subscription Design*.

Figure 22–1 Item Induction



Batch Design Summary

The following batch designs are included in this functional area:

- loadods.ksh (Item Induction)
- iindbatch.ksh (Upload Item Data)
- ld_iindfiles.ksh (Upload Data From Templates)

loadods.ksh (Item Induction)

Module Name	loadods.ksh
Description	Spreadsheet Tables Upload
Functional Area	Admin
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS473
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to upload data from the template for output files to the S9T_FOLDER table. The path of template files (ODS_SYSTEM_TEMPLATE_FOR_OUTPUT_FILES.ods) is passed as input parameter to this batch.

This program will be called from other shell script ld_iindfiles.ksh which does the initial validations to check if the template file exists.

Restart/Recovery

The restart recovery is different from the conventional Merchandising batch. There are two points on the batch upload process where users can evaluate the successful load of the data.

1. SQL load - In this program control and data files are created dynamically. In case of any error while creation of data/control file a non-fatal code is returned by the program and a message will be written to the log file.
2. **Action Required:** When such conditions exist, you should check if template files passed are valid and in expected format.
3. Other Validation - At this point data from the file(s) are loaded into the staging table(s). PL/SQL function is used to get the next sequence for each file_id. In case of any error while getting next sequence value from sequence - s9t_folder_seq fatal code is returned by the program and a message will be written to the log and error file.
4. **Action Required:** When this condition exists, you needs to check for a database connection and state of sequence should be valid in the database.

I/O Specification

Integration Type	Upload to Merchandising
File Name	Determined by runtime parameter.
Integration Contract	IntCon000216

Input File Specification - SQL Loader Input File Layout

Refer to ODS_SYSTEM_TEMPLATE_FOR_OUTPUT_FILES.ods.

iindbatch.ksh (Upload Item Data)

Module Name	iindbatch.ksh
Description	Upload Item Data
Functional Area	Item Maintenance
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS474
Wrapper Script	rmswrap_shell_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to Bulk upload xml file data from template files to the Merchandising templates table.

This batch program also accepts files in simple XML format, since some of the integrating systems, such as AP, are unable to fill up all the information needed for an Open Office Spreadsheet format files (*.ods) type of XML

This batch will be responsible for validating the input parameters, below are the list of validations.

- The Input file should exist.
- The Input file's extension must be ".xml".
- The template name should be valid.
- Destination (Optional Parameter) should be STG or Merchandising. If destination is not passed then default it to STG.
- Once xml data is loaded into the staging table, the script will do the following:
  - Initializes a row in the process tracker table for asynchronous processing.
  - Call the main induction process that uploads data into the staging tables, validates and inserts data into the base Merchandising item tables.

Restart/Recovery

N/A

Key Tables Affected

Table 22–1 Key Tables Affected

Table	Select	Insert	Update	Delete
S9T_FOLDER	No	Yes	No	No
S9T_TEMPLATE	Yes	No	No	No
SVC_PROCESS_TRACKER	No	Yes	No	No
SVC_ITEM_MASTER	Yes	Yes	Yes	Yes
SVC_ITEM_SUPPLIER	Yes	Yes	Yes	Yes
SVC_ITEM_SUPP_ COUNTRY	Yes	Yes	Yes	Yes
SVC_ITEM_SUPP_ COUNTRY_DIM	Yes	Yes	Yes	Yes
SVC_ITEM_SUPP_MANU_ COUNTRY	Yes	Yes	Yes	Yes
SVC_ITEM_SUPP_UOM	Yes	Yes	Yes	Yes
SVC_ITEM_XFORM_HEAD	Yes	Yes	Yes	Yes
SVC_ITEM_XFORM_ DETAIL	Yes	Yes	Yes	Yes
SVC_ PACKITEM	Yes	Yes	Yes	Yes
SVC_VAT_ITEM	Yes	Yes	Yes	Yes
SVC_UDA_ITEM_FF	Yes	Yes	Yes	Yes
SVC_UDA_ITEM_DATE	Yes	Yes	Yes	Yes
SVC_ITEM_IMAGE	Yes	Yes	Yes	Yes
SVC_ITEM_MASTER_TL	Yes	Yes	Yes	Yes

Table 22–1 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_ITEM_SUPPLIER_TL	Yes	Yes	Yes	Yes
SVC_ITEM_IMAGE_TL	Yes	Yes	Yes	Yes
SVC_ITEM_HTS	Yes	Yes	Yes	Yes
SVC_ITEM_HTS_ASSESS	Yes	Yes	Yes	Yes
SVC_ITEM_EXPENSES	Yes	Yes	Yes	Yes
SVC_ITEM_TICKET	Yes	Yes	Yes	Yes
SVC_ITEM_SEASONS	Yes	Yes	Yes	Yes
SVC_ITEM_CHRG	Yes	Yes	Yes	Yes
SVC_ITEM_XFORM_ HEAD_TL	Yes	Yes	Yes	Yes
CORESVC_ITEM_ERR	Yes	Yes	No	No
S9T_ERRORS	Yes	Yes	No	No
ITEM_MASTER	Yes	Yes	Yes	Yes
ITEM_SUPPLIER	Yes	Yes	Yes	Yes
ITEM_SUPP_COUNTRY	Yes	Yes	Yes	Yes
ITEM_SUPP_COUNTRY_ DIM	Yes	Yes	Yes	Yes
ITEM_SUPP_MANU_ COUNTRY	Yes	Yes	Yes	Yes
ITEM_SUPP_UOM	Yes	Yes	Yes	Yes
ITEM_XFORM_HEAD	Yes	Yes	Yes	Yes
ITEM_XFORM_DETAIL	Yes	Yes	Yes	Yes
PACKITEM	Yes	Yes	Yes	Yes
VAT_ITEM	Yes	Yes	Yes	Yes
UDA_ITEM_FF	Yes	Yes	Yes	Yes
UDA_ITEM_DATE	Yes	Yes	Yes	Yes
ITEM_IMAGE	Yes	Yes	Yes	Yes
ITEM_MASTER_TL	Yes	Yes	Yes	Yes
ITEM_SUPPLIER_TL	Yes	Yes	Yes	Yes
ITEM_IMAGE_TL	Yes	Yes	Yes	Yes
ITEM_HTS	Yes	Yes	Yes	Yes
ITEM_HTS_ASSESS	Yes	Yes	Yes	Yes
ITEM_EXP_HEAD	Yes	Yes	Yes	Yes
ITEM_EXP_DETAIL	Yes	Yes	Yes	Yes
ITEM_TICKET	Yes	Yes	Yes	Yes
ITEM_SEASONS	Yes	Yes	Yes	Yes
ITEM_CHRG_HEAD	Yes	Yes	Yes	Yes
ITEM_CHRG_DETAIL	Yes	Yes	Yes	Yes

Table 22–1 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_XFORM_HEAD_TL	Yes	Yes	Yes	Yes

Design Assumptions

N/A

ld_iindfiles.ksh (Upload Data From Templates)

Module Name	ld_iindfiles.ksh
Description	Updload Data From Templates
Functional Area	Item Maintenance
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS199
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program is used to upload data from template files to the Merchandising staging table calling another script loadods.ksh. Once data is loaded into the staging table it will do post processing, uploading data to other spreadsheet tables. This program will be responsible for validating if input files (ODS_SYSTEM_TEMPLATE_FOR_OUTPUT_FILES.ods and template_config.ods) are present in input directory (passed as parameter).

Note: There is no wrapper script for this batch program. It is invoked directly by the installer.

Restart/Recovery

N/A

Key Tables Affected

Table 22–2 Key Tables Affected

Table	Select	Insert	Update	Delete
S9T_FOLDER	No	Yes	No	Yes
S9T_TMPL_COLS_DEF_TL	No	Yes	No	Yes
S9T_TMPL_COLS_DEF	No	Yes	No	Yes

Table 22–2 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
S9T_TMPL_WKSHT_DEF_TL	No	Yes	No	Yes
S9T_TMPL_WKSHT_DEF	No	Yes	No	Yes
S9T_TEMPLATE_TL	No	Yes	No	Yes
S9T_TEMPLATE	No	Yes	No	Yes

Design Assumptions

N/A

itm_indctn_purge (Purge Item Induction Staging Tables)

Module Name	itm_indctn_purge.ksh
Description	Purge item induction staging tables
Functional Area	Foundation - Items
Module Type	Admin
Module Technology	Shell Script
Catalog ID	RMS498
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to remove old item records from the staging tables. Records that are candidates for deletion are:

- Processes that have successfully been processed or processed with warnings that have been uploaded to Merchandising or downloaded to S9T
- Processes that have a status of processed with errors and have no linked data
- Processes in error status where all other related records containing the process ID have been processed successfully
- Processes that have errors and are past the data retention days
- All item records within a process where all related records for the item in the other staging tables are successfully uploaded to Merchandising. The process tracker record for that process should not be deleted if there are other item records that are not uploaded to Merchandising.

Restart/Recovery

Restartability is implied, because the records that are selected from the cursor are deleted before the commit.

Key Tables Affected

Table 22–3 Key Tables Affected

Table	Select	Insert	Update	Delete
PROC_DATA_ RETENTION_DAYS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
SVC_PROCESS_TRACKER	Yes	No	No	Yes
SVC_PROCESS_ITEMS	No	No	No	Yes
SVC_ITEM_COST_DETAIL	No	No	No	Yes
SVC_ITEM_COST_HEAD	No	No	No	Yes
SVC_ITEM_COUNTRY	No	No	No	Yes
SVC_ITEM_COUNTRY_ L10N_EXT	No	No	No	Yes
SVC_ITEM_MASTER	No	No	No	Yes
SVC_ITEM_MASTER_TL	No	No	No	Yes
SVC_ITEM_MASTER_CFA_ EXT	No	No	No	Yes
SVC_ITEM_SUPPLIER	No	No	No	Yes
SVC_ITEM_SUPPLIER_TL	No	No	No	Yes
SVC_ITEM_SUPPLIER_ CFA_EXT	No	No	No	Yes
SVC_ITEM_SUPP_ COUNTRY	No	No	No	Yes
SVC_ITEM_SUPP_ COUNTRY_CFA_EXT	No	No	No	Yes
SVC_ITEM_SUPP_ COUNTRY_DIM	No	No	No	Yes
SVC_ITEM_SUPP_MANU_ COUNTRY	No	No	No	Yes
SVC_ITEM_SUPP_UOM	No	No	No	Yes
SVC_ITEM_XFORM_ DETAIL	No	No	No	Yes
SVC_ITEM_XFORM_HEAD	No	No	No	Yes
SVC_ITEM_XFORM_ HEAD_TL	No	No	No	Yes
SVC_PACKITEM	No	No	No	Yes
SVC_RPM_ITEM_ZONE_ PRICE	No	No	No	Yes
SVC_XITEM_RIZP_LOCS	No	No	No	Yes
SVC_XITEM_RIZP	No	No	No	Yes
SVC_ITEM_SEASONS	No	No	No	Yes
SVC_UDA_ITEM_DATE	No	No	No	Yes
SVC_UDA_ITEM_FF	No	No	No	Yes

Table 22–3 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SVC_UDA_ITEM_LOV	No	No	No	Yes
SVC_VAT_ITEM	No	No	No	Yes
SVC_ITEM_IMAGE	No	No	No	Yes
SVC_ITEM_IMAGE_TL	No	No	No	Yes
SVC_ITEM_HTS	No	No	No	Yes
SVC_ITEM_HTS_ASSESS	No	No	No	Yes
SVC_ITEM_EXPENSES	No	No	No	Yes
SVC_ITEM_TICKET	No	No	No	Yes
SVC_COST_SUSP_SUP_HEAD	No	No	No	Yes
SVC_COST_SUSP_SUP_DETAIL_LOC	No	No	No	Yes
SVC_COST_SUSP_SUP_DETAIL	No	No	No	Yes
SVC_CFA_EXT	No	No	No	Yes
CORESVC_ITEM_ERR	No	No	No	Yes
S9T_ERRORS	No	No	No	Yes
SVC_PROCESS_CHUNKS	No	No	No	Yes
S9T_FOLDER	No	No	No	Yes

Design Assumptions

N/A

23

Integration with Point of Sale

This chapter contains information about the batch processes used to send foundation and item information to a POS system in stores from Merchandising. However, it should be noted that the processes outlined in this chapter are not used by the Oracle Retail Xstore Suite. For information on how Xstore POS receives information from Merchandising similar to what was outlined in this chapter, see the *Oracle Retail Xstore Suite and Merchandising Suite Implementation Guide*.

Merchandising creates files of foundation data information for a POS solution. Files can contain either:

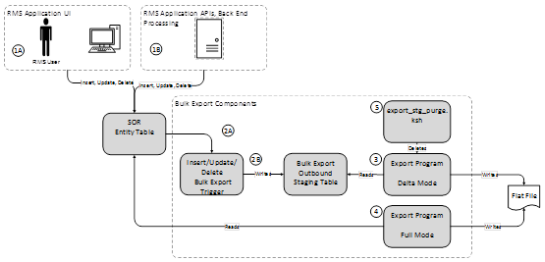
- Changes (additions, modifications, deletes) since last bulk export
- Full set of data for the entity

The goal of both forms of integration is to present complete entities to downstream systems in a neutral format. Merchandising expects the downstream systems will filter and transform the foundation data, as needed, which may include dropping some of the attributes included in the files for the various entities.

Bulk Export Pattern

There are some entity specific variations (detailed in the program specific details in this chapter), but Merchandising uses a general pattern for foundation data bulk export:

Figure 23–1 Bulk Export Pattern



Pattern Conceptual Flow:

1. (1A) Using Merchandising application UI, business user or (1B) API/Batch Process performs an insert/update/delete on a System of Record table.
2. (2A) Trigger on SOR entity table fires on insert/update/delete. (2B) Trigger writes new/changed/deleted information to outbound staging table.

3. In a delta mode, program reads bulk export staging table to get recently created, modified and deleted records and writes them to a file. Records are marked as exported.
4. In a full mode, program reads all current records from the SOR table and writes them to a file.

> **Note:** Recently deleted records are not part of the data set

5. export_stg_purge.ksh drops aged partitions from the export outbound staging tables.

> **Note:** If bulk extract programs are not run for some time, it is possible that delta records will be purges without having been exported. It is important to run these jobs daily.

Program Summary

Table 23–1 Program Summary

Program	Description
export_merchhier	Extract of Merchandise Hierarchy Data
export_orghier	Extract of Organizational Hierarchy Data
export_stores	Extract of Store Data
export_diffs	Extract of Differentiators
export_diffgrp	Extract of Diff Groups
export_itemloc	Extract of Item Locations
export_itemvat	Extract of Item VAT Data
export_itemmaster	Extract of Items
export_vat	Extract of VAT Data
export_relitem	Extract of Related Items
poscdnld	Extract of POS Configuration Data
taxdnld	Tax Download - used only when configured to use GTAX as the default tax type (Brazil only)
export_stg_purge	Purge Staged Data

It is likely that all 3rd Party POS Integration programs will not be used by most clients. The programs a client should use are dependent on their POS systems, business processes for managing those POS systems and operations requirements.

The information sent to 3rd Party POS systems falls into the following broad categories:

Table 23–2 Categories 3rd Party POS Systems

Category	Programs	Usage Recommendation
POS Configuration	poscdnld.pc	Only run this program if you use Merchandising to master POS Copnfiguration Data
Foundation	export_merhhier.ksh	This program sends merchandise hierarchy information to POS systems. Client business process and POS system requirements will determine if this program needs to be run. See the Chapter in this guide regarding Xstore integration.
Tax	taxdnld.pc	This program should be used when clients run 'GTAX' Tax

export_merchhier.ksh (Extract of Merchandise Hierarchy data)

Module Name	export_merchhier.ksh
Description	Extraction of merchandise hierarchy data.
Functional Area	Foundation
Module Type	Integration
Module Technology	ksh
Catalog ID	260
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising merchandise hierarchy information from division to subclass into a flat file. Data to be extracted will be pulled off from the merchandise hierarchy export staging table and the main merchandise hierarchy tables.

The mode (full vs. delta) will be an input parameter for this new batch. The mode will allow a full extract (all merchandise hierarchy records in Merchandising) as well as delta processing (all merchandise hierarchy changes since the last export) of data.

For a full extract, records will be solely retrieved from the main merchandise hierarchy tables. For a delta extract, the action type and entity ID will be retrieved from the merchandise hierarchy export staging table and the attributes of the entities will be retrieved from their corresponding man entity tables.

Restart/Recovery

N/A

Key Tables Affected

Table 23–3 Key Tables Affected

Table	Select	Insert	Update	Delete
MERCHHIER_EXPORT_STG	Yes	No	Yes	No
COMPHEAD	Yes	No	No	No
DIVISION	Yes	No	No	No
GROUPS	Yes	No	No	No
DEPS	Yes	No	No	No
CLASS	Yes	No	No	No
SUBCLASS	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type Extract from Merchandising

File Name merchhierarchy_[Date]_[full/delta]_[#ofLines].dat

Integration Contract IntCon000207

Design Assumptions

N/A

export_orghier.ksh (Extract of Organizational Hierarchy Data)

Module Name export_orghier.ksh

Description Extraction of organizational hierarchy data.

Functional Area Foundation

Module Type Integration

Module Technology Ksh

Catalog ID RMS261

Wrapper Script rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising organizational hierarchy information from company to stores and warehouses into a flat file. Data to be extracted will be pulled off from the organizational hierarchy export staging table and the main organizational hierarchy tables.

The mode (full vs. delta) will be an input parameter for this batch. The mode will allow a full extract (all organizational hierarchy records in Merchandising) as well as delta processing (all organizational hierarchy changes since the last export) of data.

For a full extract, records will be solely retrieved from the main organizational hierarchy tables. For a delta extract, the action type and entity ID will be retrieved from the organizational hierarchy export staging table and the attributes of the entities will be retrieved from their corresponding man entity tables.

Restart/Recovery

N/A

Key Tables Affected

Table 23–4 Key Tables Affected

Table	Select	Insert	Update	Delete
ORGHIER_EXPORT_STG	Yes	No	Yes	No
COMPHEAD	Yes	No	No	No
CHAIN	Yes	No	No	No
AREA	Yes	No	No	No
REGION	Yes	No	No	No
DISTRICT	Yes	No	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type Extract from Merchandising

File Name orghierarchy_[Date]_[full/delta]_[#ofLines].dat

Integration Contract IntCon000203

Design Assumptions

N/A

export_stores.ksh (Extract of Store Data)

Module Name export_stores.ksh

Description	Extraction of store data.
Functional Area	Foundation
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS263
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising store information into two flat files - one for store and one for store addresses. Data to be extracted will be pulled from the store export staging, store and address tables.

The mode (full vs. delta) will be an input parameter for this batch. The mode will allow a full extract (all store records in Merchandising) as well as delta processing (all store changes since the last export) of data.

For a full extract, records will be solely retrieved from the store table for store information and address table for store addresses. For a delta extract, the action type, store ID and address will be retrieved from the store export staging table and the details of the store will be retrieved from both the store and address tables.

Restart/Recovery

N/A

Key Tables Affected

Table 23–5 Key Tables Affected

Table	Select	Insert	Update	Delete
STORE_EXPORT_STG	Yes	No	Yes	No
STORE	Yes	No	No	No
ADDR	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type	Extract from Merchandising
File Name	store_[Date]_[full/delta]_[#ofLines].dat
	storeaddr_[Date]_[full/delta]_[#ofLines].dat
Integration Contract	IntCon000204
	IntCon000205

Design Assumptions

N/A

export_diffs.ksh (Extraction of differentiators data defined for a differentiator type)

Module Name	export_diffs.ksh
Description	Extraction of differentiator's data defined for a differentiator type.
Functional Area	Foundation
Module Type	Integration
Module Technology	ksh
Catalog ID	256
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising differentiator information into a flat file. Data to be extracted will be pulled off from the differentiator extract staging and the differentiator IDs table.

The mode (full vs. delta) will be an input parameter for this batch. The mode will allow a full extract (all differentiator records in Merchandising) as well as delta processing (all differentiator record changes in the time frame passed in the program) of data.

For a full extract, records will be solely retrieved from the differentiator IDs table. For a delta extract, the action type and differentiator ID will be retrieved from the differentiator export staging table and the attributes will be retrieved from the differentiator IDs table.

Restart/Recovery

N/A

Key Tables Affected

Table 23–6 Key Tables Affected

Table	Select	Insert	Update	Delete
DIFFS_EXPORT_STG	Yes	No	Yes	No
DIFF_IDS	Yes	No	No	No
DIFF_TYPE	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type	Extract from Merchandising
File Name	diffs_date_[full/delta]_[#ofLines].dat
Integration Contract	IntCon000206.html

Design Assumptions

N/A

export_diffgrp.ksh (Extraction of differentiator groups data)

Module Name	export_diffgrp.ksh
Description	Extraction of differentiator groups data.
Functional Area	Foundation
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS255
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising diff group information into a flat file. Data to be extracted will be pulled off from the differentiator group tables.

The mode (full vs. delta) will be an input parameter for this batch. The mode will allow a full extract (all diff group records in Merchandising) as well as delta processing (all diff group record changes in the time frame passed in the program) of data.

For a full extract, records will be retrieved from the differentiator group tables. For a delta extract, the action type and diff group ID will be retrieved from the differentiator group staging export table and the attributes will be retrieved from the differentiator group tables.

Restart/Recovery

N/A

Key Tables Affected

Table 23–7 Key Tables Affected

Table	Select	Insert	Update	Delete
DIFFGRP_EXPORT_STG	Yes	No	Yes	No

Table 23–7 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
DIFF_GROUP_HEAD	Yes	No	No	No
DIFF_GROUP_DETAIL	Yes	No	No	No
DIFF_IDS	Yes	No	No	No
DIFF_TYPE	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type	Extract from Merchandising
File Name	diffgrphdr_date_[full/delta]_[#ofLines].dat diffgrpdtl_date_[full/delta]_[#ofLines].dat
Integration Contract	IntCon000212.html IntCon000213.html

Design Assumptions

N/A

export_itemloc.ksh (Extraction of item location data)

Module Name	export_itemloc.ksh
Description	Extraction of item location data.
Functional Area	Foundation
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS257
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job extracts new, updated and deleted Merchandising item-location information into a flat file.

- This batch supports both a full and delta export of item-location data.
- A threading indicator parameter should be passed. Passing 'Y' means a thread number (1-20) will be passed in. Passing 'N' means no thread number will be passed in and the program will use a default thread number.

- An optional location parameter may be passed in for either modes. If this value is passed in, the batch will create a flat file for the location passed in. If it is not passed in, the batch will create flat files for all locations.
- This creates separate files per location (Store, Warehouse or External Finisher).
- This exports delta item header information for each applicable store location.
- This will export data only for approved, sellable items.
- This will export attributes from item/location and item/location traits.
- This should also include the item parent as its own record in the extract.
- The flat files that will be created will now be pipe delimited.

Restart/Recovery

N/A

Key Tables Affected

Table 23–8 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_EXPORT_INFO	Yes	No	No	No
ITEM_EXPORT_STG	Yes	No	Yes	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC_TRAITS	Yes	No	No	No
STORE	Yes	No	No	No
WH	Yes	No	No	No
PARTNER	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type	Extract from Merchandising
File Name	itemloc_[#date]_[#loc_type]_[#location]_[full/delta]_[#ofLines].dat
	itemhdr_[#date]_[#store_id]_delta_[#ofLines].dat
Integration Contract	IntCon000209
	IntCon000208

Design Assumptions

N/A

export_itemvat.ksh (Extraction of vat item data)

Module Name	export_itemvat.ksh
Description	Extraction of vat item data.
Functional Area	Foundation
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS259
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising item VAT information into a flat file.

- This batch supports both a full and delta export of item VAT data.
- A threading indicator parameter should be passed. Passing 'Y' means a thread number (1-20) will be passed in. Passing 'N' means no thread number will be passed in and the program will use a default thread number.
- In full mode, normal operation will produce both a corporate level file and files for all stores. An optional input parameter will also allow the program to produce a location level file for a specified store.
- In full mode for store specific file if store belong to such a vat region, which is exempt (In case of tax type SVAT), then files for that store won't get generated.
- In delta mode, this will produce both corporate level files and files for all stores the modified items are ranged to and the vat region the store is associated with.
- In delta mode for store specific file if store belong to such a vat region, which is exempt, then files for that store won't get generated.
- This will export data only for approved, sellable items.
- This will export item VAT information from the item export staging and item tables.
- This should also include the item parent as its own record in the extract.
- The flat files that will be created will now be pipe delimited.

Restart/Recovery

N/A

Key Tables Affected

Table 23–9 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_EXPORT_INFO	Yes	No	No	No
ITEM_EXPORT_STG	Yes	No	Yes	No
VAT_REGION	Yes	No	No	No
VAT_ITEM	Yes	No	No	No
STORE	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type Extract from Merchandising

File Name vatitem_[#date]_corp_[full/delta]_[#ofLines].dat

vatitem_[#date]_[location]_[full/delta]_[#ofLines].dat

Integration Contract IntCon000214

Design Assumptions

N/A

export_itemmaster.ksh (Extraction of item data)

Module Name export_itemmaster.ksh

Description Extraction of item data

Functional Area Foundation

Module Type Integration

Module Technology ksh

Catalog ID RMS258

Wrapper Script rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising item master information into a flat file.

Data to be extracted will be pulled off from the item export information, item export staging and the main item tables.

- The mode (full vs. delta) will be an input parameter for this new batch. The mode will allow a full extract (all approved, sellable items in Merchandising) as well as delta processing (all approved, sellable item changes in the main item table since the last export) of data.
- A threading indicator parameter should be passed. Passing 'Y' means a thread number (1-20) will be passed in. Passing 'N' means no thread number will be passed in and the program will use a default thread number.
- In full mode, normal operation will produce both a corporate level file and files for all stores. An optional input parameter will also allow the program to produce a location level file for a specified store.
- In delta mode, the only option is to produce corporate level files. Item header files at the store level will be created in the export_itemloc.ksh for delta mode.
- The store specific file will also include UPC items. To determine which UPC Items to include, the store where the UPC's parent and/or grandparent item is ranged should be taken into consideration.
- The flat files that will be created will now be pipe delimited.

Restart/Recovery

N/A

Key Tables Affected

Table 23–10 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_EXPORT_INFO	Yes	No	Yes	No
ITEM_EXPORT_STG	Yes	No	Yes	No
CLASS	Yes	No	No	No
SUBCLASS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
DIFF_IDS	Yes	No	No	No
DIFF_GROUP_HEAD	Yes	No	No	No
STORE	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type Extract from Merchandising

File Name itemhdr_[#date]_corp_[full/delta]_[#ofLines].dat

itemhdr_[#date]_[location]_full_[#ofLines].dat

Integration Contract	IntCon000208

Design Assumptions

N/A

export_vat.ksh (Extraction of vat data)

Module Name	export_vat.ksh
Description	Extraction of vat data
Functional Area	Foundation
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS264
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising VAT information into a flat file. Data to be extracted will be pulled off from the VAT export staging, VAT region, VAT codes, and VAT code rates tables.

The mode (full vs. delta) will be an input parameter for this batch. The mode will allow a full extract (all vat region/vat code/vat code rate combination records in RMS) as well as delta processing (all VAT record changes in the time frame passed in the program) of data.

In either of the mode exempt vat region won't get fetched in case of SVAT tax type.

For a full extract, records will be retrieved from the VAT region, VAT code, and VAT code rates tables. For a delta extract, the action type, vat region, vat code and active date will be retrieved from the VAT export staging table and the attributes will be retrieved from the main table.

Restart/Recovery

N/A

Key Tables Affected

Table 23–11 Key Tables Affected

Table	Select	Insert	Update	Delete
VAT_EXPORT_STG	Yes	No	Yes	No
VAT_CODES	Yes	No	No	No

Table 23–11 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
VAT_CODE_RATES	Yes	No	No	No
VAT_REGION	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type Extract from Merchandising

File Name vat_date_[full/delta]_[#ofLines].dat

Integration Contract IntCon000215

Design Assumptions

N/A

export_relitem.ksh (Extraction of item data)

Module Name export_relitem.ksh

Description Extraction of related item data

Functional Area Foundation

Module Type Integration

Module Technology ksh

Catalog ID RMS262

Wrapper Script rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract new, updated and deleted Merchandising related items information into a flat file.

- This batch will support both a full and delta export of related item data.
- A threading indicator parameter should be passed. Passing 'Y' means a thread number (1-20) will be passed in. Passing 'N' means no thread number will be passed in and the program will use a default thread number.
- In full mode, normal operation will produce both a corporate level files and files for all stores. An optional input parameter will also allow the program to produce location level files for a specified store.
- In delta mode, this will produce both corporate level files and files for all stores the modified data are ranged to.
- This will export data only for approved, sellable items.

- This will export item related item information from the related item export staging and related item tables.
- Two types of flat files will be created for this extract - one for the related item header information and one for the related item detail information.
- When creating the location level files, ensure that both items (the main item and related item) are ranged in the location.
- The flat files that will be created will now be pipe delimited.

Restart/Recovery

N/A

Key Tables Affected

Table 23–12 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_EXPORT_INFO	Yes	No	No	No
RELITEM_EXPORT_STG	Yes	No	Yes	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
RELATED_ITEM_HEAD	Yes	No	No	No
RELATED_ITEM_DETAIL	Yes	No	No	No
STORE	Yes	No	No	No
DATA_EXPORT_HIST	No	Yes	No	No

I/O Specification

Integration Type Extract from Merchandising

File Name relitemhead_date_corp_[full/delta]_[#ofLines].dat

relitemhead_date_[Location]_[full/delta]_[#ofLines].dat

relitemdet_date_corp_[full/delta]_[#ofLines].dat

relitemdet_date_[Location]_[full/delta]_[#ofLines].dat

Integration Contract IntCon000210

IntCon000211

Design Assumptions

N/A

taxdnld (Tax Download to 3rd Party POS in Global Tax [GTAX] Implementations)

Module Name taxdnld

Description	Tax Download to 3rd Party POS in Global Tax [GTAX] Implementations
Functional Area	Integration - 3rd Party POS
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS124
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program downloads the tax information to 3rd Party POS systems when the Merchandising default tax type is GTAX.

This program only needs to be run if the client uses Merchandising Global Tax functionality.

Restart/Recovery

The logical unit of work for this module is defined by item, ref_item and store combination. This batch program uses table-based restart/recovery. The commit happens in the database when the commit max counter is reached.

Key Tables Affected

Table 23–13 Key Tables Affected

Table	Select	Insert	Update	Delete
POS_MODS_TAX_INFO	Yes	No	No	No
GTAX_ITEM	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
V_RESTART_STORE	Yes	No	No	No
CLASS	Yes	No	No	No

Integration Contract

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000020

Output File Layout

Table 23–14 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	File Type Definition	Char(4)	TAXD	Identifies file as 'Tax Details'
	File Create Date	Char(14)	create date	Vdate in 'YYYMMDDHHMISS'format
FDETL	FDETL	Char(5)	FDETL	FDETL
	File Line Sequence	Number(10)	N/A	Line number of the current file
	STORE	Char(10)	N/A	Store number
	ITEM	Char(25)	N/A	Item
	item_number_type	Char(6)	S - Store W - Warehouse	Item number type
	format_id	Char(1)	N/A	Format id
	prefix	Char(2)	N/A	Prefix
	ref_item	Char(25)	N/A	Reference Item
	ref_item_number_type	Char(6)	N/A	Refrerence item number type
	ref_format_id	Char(1)	N/A	Ref format id
	ref_prefix	Char(2)	N/A	Ref no. prefix
	taxable indicator	Char(1)	N/A	Taxable indicator
	class_vat_ind	Char(1)	N/A	Class vat indicator

Table 23–14 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FTAXD	FTAXD	Char(5)	FTAXD	FTAXD
	File Line Sequence	Number(10)	N/A	Line number of the current file
	tax_code	Char(10)	N/A	Tax code
	tax_rate	Char(20)	N/A	Tax rate
	calculation_ basis	Char(1)	N/A	Calculation basis
	tax_amount	Char(20)	N/A	Tax amount
	effective_ from	Char(8)	N/A	Effective from
	time	Char(6)	N/A	Time
	status	Char(1)	N/A	Status
FTAIL	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type
	File Line Sequence	Number(10)	N/A	Line number of the current file
	rec_counter	Number(10)	N/A	Record counter

Design Assumptions

N/A

poscdnld (Download of POS Configuration Data to 3rd Party POS)

Module Name	poscdnld.pc
Description	Download of POS Configuration Data to 3rd Party POS
Functional Area	Integration - 3rd Party POS
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS69
Wrapper Script	rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program downloads POS configuration information from Merchandising to a flat file. This file can be used to load POS and back-office systems.

This program (and its related prepost function) should only be run if Merchandising is used to master:

- Coupon definitions and relationships to items
- Restrictions on product sales, including but not limited to minimum age of purchaser, time/days when product cannot be sold, tenders that cannot be used to purchase the product, and so on.

Restart/Recovery

The logic unit of work is pos configuration type and pos configuration ID. The commit_max_ctr field should be set to prevent excessive rollback space usage, and to reduce the overhead of file I/O. The recommended commit counter setting is 1000 records (subject to change based on implementation).

Key Tables Affected

Table 23–15 Key Tables Affected

Table	Select	Insert	Update	Delete
POS_PROD_REST_HEAD	Yes	No	No	Yes
POS_COUPON_HEAD	Yes	No	No	Yes
POS_CONFIG_ITEMS	Yes	No	No	Yes
POS_STORE	Yes	No	No	Yes
POS_DAY_TIME_DATE	Yes	No	No	Yes
POS_MERCH_CRITERIA	Yes	No	No	Yes
DEPS	Yes	No	No	No
ITEM_LOC	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
PERIOD	Yes	No	No	No

I/O Specification

Integration Type Download from Merchandising

File Name Determined by runtime parameter.

Integration Contract IntCon000063

Output File Layout

Table 23–16 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record Type	Char(5)	'FHEAD'	Record Identifier
	Line id	Number(10)	0000000001	Sequential Line Identifier
	File Name	Char(4)	'POSC'	File Identifier
	File Date	Char(14)	N/A	Date the file was created in 'YYYYMMDD HHMMSS' format
TCOUP	Record Type	Char(5)	TCOUP	Record Identifier
	Line id	Number(10)	N/A	Sequential Line Identifier
	Coupon id	Number(6)	N/A	N/A
	Coupon Desc	Char(250)	N/A	N/A
	Currency Code	Char2(3)	N/A	N/A

Table 23–16 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Max Discount Amount	Number(20,4)	N/A	N/A
	Amount	Number(20,4)	N/A	N/A
	Percent Ind	Char(1)	'N' - Amount 'Y'- Percentage	N/A
	Profit Center	Char(6)	N/A	N/A
	Tax Class	Char(6)	N/A	N/A
	Export Code	Char(6)	N/A	N/A
	Effective Date	Char(14)	N/A	Indicates the first day the coupon can be used in 'YYYYMMDD HHMMSS' format
	Expiration Date	Char(14)	N/A	Indicates the day the coupon becomes invalid in 'YYYYMMDD HHMMSS' format
	Prompted Ind	Char(1)	'Y', 'N'	This indicator identifies if the cashier should be prompted to ask for a Coupon.
	Display Ind	Char(1)	'Y', 'N'	This indicator specifies whether the coupon is displayed in the list of valid coupons on the register.
	Status	Char(1)	'A','C','D'	Indicates if the coupon configuration is new, has been changed, or being deleted.
	Vendor	Number(10)	N/A	N/A

Table 23–16 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Vendor Type	Char(6)	'AG' - Agent 'AP' - Applicant 'BK' - Bank 'BR' - Broker 'CN' - Coonsignee 'CO' - Consolidator 'FA' - Factory 'FF' - Freight Forwarder 'IM' - Importer 'SU' - Supplier	N/A
	Promotion	Number(10)	N/A	N/A
	Coupon Barcode	Char(20)	N/A	N/A
	Coupon Max Qty	Number(6)	N/A	N/A

Table 23–16 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TPRES	Record Type	Char(5)	TPRES	Record Identifier
	Line id	Number(10)	N/A	Sequential Line Identifier
	POS Product Restriction id	Number(6)	N/A	N/A
	POS Product Restriction Desc	Char(120)	N/A	N/A
	POS Product Restriction Type	Char(6)	'PPRT' include: 'STMP' - Food Stamp 'MNAG' - Minimum Age 'CNDP' -Container Deposit 'CNVL' - Container Redemption Value 'DTDR' - Day/Time/ Date Restriction 'TENT' - Tender Type 'NDSC' - Non-Discoun table 'RTRN' - Returnable 'QLMT' - Quantity Limit	N/A
	Effective Date	Char(14)	N/A	Date the product restriction is first effective in 'YYYYMMDD HHMMSS' format
	Currency Code	Char(3)	N/A	N/A
	Product Restriction Amount	Number(20,4)	N/A	N/A

Table 23–16 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Age Minimum	Number(2)	N/A	N/A
	Date Restriction	Char(14)	N/A	Date on which a specified product restriction is applied in 'YYYYMMDD HHMMSS' format
	Before Time Restriction	Char(6)	N/A	N/A
	After Time Restriction	Char(6)	N/A	N/A
	Day Restriction	Char(6)	N/A	N/A
	Max Qty Amount	Number(12,4)	N/A	N/A
	Tender Type Group	Char(6)	'CASH' - Cash, 'CHECK' - Check, 'CCARD' - Credit, 'COUPON' - Coupon, 'LOTTRY' - Lottery, 'FSTAMP' - Food Stamp, 'DCARD' - Debit Card, 'MORDER' - Money Order 'VOUCH' - Voucher 'ERR' - Error, 'SOCASS' - Social Assistance, 'TERM' - Termination Record, 'DRIVEO' - Drive Off, 'EBS' - Electronic Benefits (Food Stamps)	N/A

Table 23–16 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Status	Char(1)	'A','C','D'	Indicates if the product restriction configuration is new, has been changed, or being deleted.
TSTOR	Record Type	Char(5)	'TSTOR'	N/A
	Line id	Number(10)	N/A	N/A
	Store	Number(10)	N/A	N/A
	Status	Char(1)	'A' - Add 'D' - Delete 'C' - Change	N/A
TITEM	Record Type	Char(5)	TITEM	Record Identifier
	Line id	Number(10)	N/A	Sequential Line Identifier
	Item	Char(25)	N/A	Left-Justified Item Identifier
	Status	Char(1)	'A' - Add 'D' - Delete 'C' - Change	Indicates the item's status at the POS. Overlays of items as a result of a change to the merch criteria will have a 'C' status.
FTAIL	Record Type	Char(5)	FTAIL	Marks end of file
	Line id	Number(10)	N/A	Total number of lines in file
	Number of transactions	Number(10)	N/A	Number of transactions in file

Design Assumptions

N/A

export_stg_purge.ksh (Purging of all the extracted data)

Module Name export_stg_purge.ksh

Description Purging of all the extracted records (week old) for Xstore.

Functional Area Foundation

Module Type Integration

Module Technology	Ksh
Catalog ID	RMS265
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will be used to remove records that are a week old from the following staging tables.

- Merchandise Hierarchy Export Staging
- Organizational Hierarchy Export Staging
- Store Export Staging
- Differentiator Export Staging
- Differentiator Group Export Staging
- Item Export Staging
- VAT Export Staging
- Related Item Export Staging
- Data Export History

Batch will purge all the records (Week old records) from its respective staging table whether data get extracted or not.

Restart/Recovery

N/A

Key Tables Affected

Table 23–17 Key Tables Affected

Table	Select	Insert	Update	Delete
MERCHHIER_EXPORT_STG	No	No	No	Yes
ORGHIER_EXPORT_STG	No	No	No	Yes
STORE_EXPORT_STG	No	No	No	Yes
DIFFS_EXPORT_STG	No	No	No	Yes
DIFFGRP_EXPORT_STG	No	No	No	Yes
ITEM_EXPORT_STG	No	No	No	Yes
VAT_EXPORT_STG	No	No	No	Yes
RELITEM_EXPORT_STG	No	No	No	Yes
DATA_EXPORT_HIST	No	No	No	Yes

Design Assumptions

N/A

24

Integration with Advanced Inventory Planning

This chapter contains information about the processes that enables packaged integration with Oracle Retail Advanced Inventory Planning (AIP).

AIP is a replenishment system. AIP uses foundation and inventory information mastered in Merchandising to suggest purchase orders. These suggested purchase orders are sent to Merchandising to be actualized.

Extracts from Merchandising are performed via batch ReTL (Retail Extract Transform) scripts described in this chapter. Suggested purchase orders are publiched to the RIB by AIP; Merchandising subscribes to these purchase order RIB messages. For more information about the PO Subscription, see the *Oracle Retail Merchandising Foundation Cloud Service Operations Guide, Volume 2 - Message Publication and Subscription Design*

According to RRA, there is one RPAS program that should be run for AIP integration:

- rmse_rpas_dailt_sales.ksh

For more information about the RPAS program, see the Integration with Oracle Retail Planning and Forecasting chapter.

Merchandising and AIP integration stands independent of additional RPAS integration for other RPAS based solutions.

AIP integration jobs only need to be scheduled if a client integrates with AIP.

Foundation Data vs Transaction/Inventory Data

AIP requires both foundation and transaction data from Merchandising. In most cases, foundation data extracts can be run ad hoc at any time.

Transaction and inventory extracts should be scheduled after main Merchandising inventory processing.

Scheduling and dependency information for each program can be found in the program details section of this chapter.

Program Summary

Table 24–1 Program Summary

Program	Description
rmse_aip_batch.ksh	Optional Wrapper Script to run all AIP Extracts
pre_rmse_aip.ksh	Extract of Merchandising System level settings for AIP

Table 24–1 (Cont.) Program Summary

Program	Description
rmse_aip_merchhier.ksh	Extract of Merchandise Hierarchy for AIP
rmse_aip_orghier.ksh	Extract of Organization Hierarchy for AIP
rmse_aip_item_master.ksh	Extract of Items for AIP
rmse_aip_store.ksh	Extract of Stores for AIP
rmse_aip_wh.ksh	Extract of Warehouses for AIP
rmse_aip_substitute_items.ksh	Extract of Substitute Items for AIP
rmse_aip_suppliers.ksh	Extract of Suppliers for AIP
rmse_aip_alloc_in_well.ksh	Extract of Allocations in the Well Quantities for AIP
rmse_aip_cl_po.ksh	Extract of AIP Generated POs, Allocations and Transfers Cancelled or Closed in Merchandising for AIP
rmse_aip_future_delivery_alloc.ksh	Extract of Allocation Quantities for Future Delivery for AIP
rmse_aip_future_delivery_order.ksh	Extract of Purchase Order Quantities for Future Delivery for AIP
rmse_aip_future_delivery_tsf.ksh	Extract On Order and In Transit Transfer Quantities for Future Delivery for AIP
rmse_aip_future_item_loc_traits.ksh	Extract of Shelf Life on Receipt Location Trait for AIP
rmse_aip_item_retail.ksh	Extract of Forecasted Items for AIP
rmse_aip_item_sale.ksh	Extract of Scheduled Item Maintenance On/Off Sale Information for AIP
rmse_aip_item_supp_country.ksh	Extract of Order Multiples by Item/Supplier/Origin Country for AIP
rmse_aip_rec_qty.ksh	Extract of Received PO, Allocation and Transfer Quantities for AIP
rmse_aip_store_cur_inventory.ksh	Extract of Store Current Inventory data for AIP
rmse_aip_tsf_in_well.ksh	Extract of Transfer in the Well Quantities to AIP
rmse_aip_wh_cur_inventory.ksh	Extract of Warehouse Current Inventory for AIP

rmse_aip_batch (Optional Wrapper Script to run all AIP Extracts)

Module Name	rmse_aip_batch.ksh
Description	Optional Wrapper Script to run all AIP Extracts
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	ksh
Catalog ID	N/A
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The rmse_aip_batch.ksh script is an optional wrapper that runs all extracts from Merchandising for AIP.

This wrapper script assumes default input parameters for some jobs. Care should be taken to ensure that if a client uses this wrapper script, those default input parameters are either correct or updated to the correct value for the implementation.

This wrapper script also assumes that all extracts from Merchandising should be run. There are cases (detailed in the extract script specific documentation) where this might not be the case. Care should be taken to ensure that if a client uses this wrapper script, it is updated as needed to reflect the extracts appropriate to the implementation.

This wrapper script also assumes that all extracts should be run sequentially at a single point in the Merchandising batch schedule. This may or may not be the best assumption for a given implementation.

If a client chooses not to use this wrapper script, he can schedule most AIP integration jobs at ad-hoc at any time in the batch schedule. Only a few jobs have specific dependencies. Most data can be sent to AIP early in the cycle. Only a few jobs will have to wait until later in the batch schedule. Some clients find are able to start the AIP processing earlier in the schedule if they do not use this wrapper script.

If a client uses this wrapper script, no extraction for AIP will be performed until the most restrictive dependencies allow it. This may mean a delay in getting any information to AIP so its processing can begin.

The wrapper script is convenient, but may not be the right choice for all implementations.

The scripts included in this wrapper are:

- pre_rmse_aip.ksh
- rmse_aip_item_master.ksh
- rmse_aip_item_supp_country.ksh
- rmse_aip_merchhier.ksh
- rmse_aip_orghier.ksh
- rmse_aip_store.ksh
- rmse_aip_suppliers.ksh
- rmse_aip_wh.ksh
- rmse_aip_item_retail.ksh
- rmse_aip_item_loc_traits.ksh
- rmse_aip_substitute_items.ksh
- rmse_aip_store_cur_inventory.ksh
- rmse_aip_wh_cur_inventory.ksh
- rmse_aip_future_delivery_alloc.ksh
- rmse_aip_alloc_in_well.ksh

- rmse_aip_future_delivery_order.ksh
- rmse_aip_future_delivery_tsf.ksh
- rmse_aip_tsf_in_well.ksh
- rmse_aip_item_sale.ksh
- rmse_aip_cl_po.ksh
- rmse_aip_rec_qty.ksh

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

N/A

pre_rmse_aip (Extract of Merchandising System level settings for AIP)

Module Name	pre_rmse_aip.ksh
Description	Extract of Merchandising System level settings for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS159
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts assorted Merchandising system level settings to files. This module produces 14 single value output files. These files can be loaded into AIP.

Most RETL programs use schema files to describe the definition of the output files. As the files produced by this module are incredibly simple, no schema files are used.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–2 Key Tables Affected

Table	Select	Insert	Update	Delete
SYSTEM_OPTIONS	Yes	No	No	No
SYSTEM_VARIABLES	Yes	No	No	No

Table 24–2 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
RETL_EXTRACT_DATES	Yes	No	No	No
CURRENCY_RATES	Yes	No	No	No

I/O Specification

Integration Type Download from Merchandising

File Name consolidation_code.txt

Integration Contract IntCon000180

Field Name	Field Type	Required
CONSOLIDATION_CODE	Varchar2(1)	Yes

Integration Type Download from Merchandising

File Name vat_ind.txt

Integration Contract IntCon000181

Field Name	Field Type	Required
VAT_IND	Varchar2(6)	Yes

Integration Type Download from Merchandising

File Name stkldgr_vat_incl_retl_ind.txt

Integration Contract IntCon000182

Field Name	Field Type	Required
STKLDGR_VAT_INCL_RETL_IND	Varchar2(1)	Yes

Integration Type Download from Merchandising

File Name multi_currency_ind.txt

Integration Contract IntCon000183

Field Name	Field Type	Required
MULTI_CURRENCY_IND	Varchar2(1)	Yes

Integration Type Download from Merchandising

File Name prime_currency_code.txt

Integration Contract IntCon000184

Field Name	Field Type	Required
CURRENCY_CODE	Varchar2(3)	Yes

Integration Type Download from Merchandising

File Name class_level_vat_ind.txt

Integration Contract IntCon000185

Field Name	Field Type	Required
CLASS_LEVEL_VAT_IND	Varchar2(1)	Yes

Integration Type Download from Merchandising

File Name domain_level.txt

Integration Contract IntCon000186

Field Name	Field Type	Required
DOMAIN_LEVEL	Varchar2(1)	Yes

Integration Type Download from Merchandising

File Name vdate.txt

Integration Contract IntCon000187

Field Name	Field Type	Required
VDATE	Date	Yes

Integration Type Download from Merchandising

File Name next_vdate.txt

Integration Contract IntCon000188

Field Name	Field Type	Required
NEXT_VDATE	Date	Yes

Integration Type Download from Merchandising

File Name last_eom_date.txt

Integration Contract IntCon000189

Field Name	Field Type	Required
LAST_EOM_DATE	Date	Yes

Integration Type Download from Merchandising

File Name curr_bom_date.txt

Integration Contract IntCon000190

Field Name	Field Type	Required
CURR_BOM_DATE	Date	Yes

Integration Type Download from Merchandising

File Name max_backpost_days.txt

Integration Contract IntCon000191

Field Name	Field Type	Required
MAX_BACKPOST_DAYS	Date	Yes

Integration Type Download from Merchandising

File Name last_extr_closed_pot_date.txt

Integration Contract IntCon000192

Field Name	Field Type	Required
LAST_EXTR_CLOSED_POT_DATE	Date	Yes

Integration Type Download from Merchandising

File Name last_extr_received_pot_date.txt

Integration Contract IntCon000193

Field Name	Field Type	Required
LAST_EXTR_RECEIVED_POT_DATE	Date	Yes

Integration Type Download from Merchandising

File Name last_extr_received_pot_date.txt

Integration Contract IntCon000194

Field Name	Field Type	Required
LAST_EXTR_RECEIVED_POT_DATE	Date	Yes

Integration Type	Download from Merchandising
File Name	prime_exchng_rate.txt
Integration Contract	IntCon000195

Field Name	Field Type	Required
PRIME_EXCHNG_RATE	Number(20,10)	Yes

rmse_aip_merchhier (Extract of Merchandise Hierarchy for AIP)

Module Name	Rmse_aip_merchhier.ksh
Description	Extract of Merchandise Hierarchy for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS32
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising merchandise hierarchy information for integration with Oracle Retail Advanced Inventory Planning (AIP).

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000077 rmse_aip_merchhier.schema

File Layout

Table 24–3 File Layout

Field Name	Field Type	Required	Description
SUBCLASS	Integer(5)	Yes	Subclass.subclass
SUB_NAME	Char(20)	Yes	Subclass.sub_name

Table 24–3 (Cont.) File Layout

Field Name	Field Type	Required	Description
CLASS	Integer(5)	Yes	Subclass.class
CLASS_NAME	Char(20)	Yes	Class.calss_name
DEPT	Integer(5)	Yes	Class.dept
DEPT_NAME	Char(20)	Yes	Deps.dept_name
GROUP_NO	Integer(5)	Yes	Deps.Group_no
GROUP_NAME	Char(20)	Yes	Groups.group_name
DIVISION	Integer(5)	Yes	Groups.division
DIV_NAME	Char(20)	Yes	Division.div_name
COMPANY	Integer(5)	Yes	Comphead.company
CO_NAME	Char(20)	Yes	Comphead.co_name
PURCHASE_TYPE	Integer(1)	Yes	Deps.purchase_type

rmse_aip_orghier.ksh (Extract of Organization Hierarchy for AIP)

Module Name	rmse_aip_orghier.ksh
Description	Extract of Organization Hierarchy for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS26
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts from the Merchandising organizational hierarchy information for integration with Oracle Retail Advanced Inventory Planning (AIP).

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–4 Key Tables Affected

Table	Select	Insert	Update	Delete
COMPHEAD	Yes	No	No	No
CHAIN	Yes	No	No	No

Table 24–4 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
AREA	Yes	No	No	No
REGION	Yes	No	No	No
DISTRICT	Yes	No	No	No

I/O Specification

Integration Type Download from Merchandising

File Name rmse_aip_orghier.dat

Integration Contract IntCon000078

rmse_aip_orghier.schema

Output File Layout

Table 24–5 File Layout

Field Name	Field Type	Required	Description
DISTRICT	Integer(11)	No	District.district
DISTRICT_NAME	Char(20)	No	District.district_name
REGION	Integer(11)	No	Region.region
REGION_NAME	Char(20)	No	Region.region_name
AREA	Integer(11)	No	Area.area
AREA_NAME	Char(20)	No	Area.area_name
CHAIN	Integer(11)	Yes	Chain.chain
CHAIN_NAME	Char(20)	Yes	Chain.chain_name
COMPANY	Integer(5)	Yes	Comphead.company
CO_NAME	Char(20)	Yes	Comphead.co_name

rmse_aip_item_master (RMS Extract of Items for AIP)

Module Name rmse_aip_item_master.ksh

Description Extract of Items for AIP

Functional Area Integration - AIP

Module Type Integration

Module Technology Ksh

Catalog ID RMS30

Wrapper Script rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising item information for integration with Oracle Retail Advanced Inventory Planning (AIP).

Two output files are produced by this extract. One contains approved transaction-level items while the other contains purged items from the daily_purge table.

Note: Items are generally not deleted from Merchandising in a one day process (records will exist on the DAILY_PURGE table for some time). This assumption means that it is reasonable for the dlyprg program (which deleted from DAILY_PURGE) to run before this extract.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Locking Strategy

N/A

Security Considerations

N/A

Performance Considerations

N/A

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_item_master.dat
Integration Contract	IntCon000073 rmse_aip_item_master.schema

Output File Layout

Table 24–6 File Layout

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	Item_master.item
ITEM_DESC	Char(250)	Yes	Item_master.item_desc
ITEM_PARENT	Char(25)	No	Item_master.item_parent
ITEM_GRANDPARENT	Char(25)	No	Item_master.item_ grandparent
AIP_SKU	Char(25)	Yes	V_packsku_qty.item or Item_master.item
SUBCLASS	Integer(5)	Yes	Item_master.subclass

Table 24–6 (Cont.) File Layout

Field Name	Field Type	Required	Description
CLASS	Integer(5)	Yes	Item_master.class
DEPT	Integer(5)	Yes	Item_master.dept
FORECAST_IND	Char(1)	Yes	Item_master.forecast_ind
SUPPLIER	Integer(11)	Yes	Item_supplier.supplier
PRIMARY_SUPP_IND	Char(1)	Yes	Item_supplier.primary_ supp_ind
STANDARD_UOM	Char(4)	Yes	Item_master.standard_uom
STANDARD_UOM_ DESCRIPTION	Char(120)	Yes	Uom_class.uom_desc
SKU_TYPE	Char(6)	No	Item_master.handling_temp or 0
SKU_TYPE_DESCRIPTION	Char(40)	No	Code_detail.code_desc (for code_type 'HTMP')
PACK_QUANTITY	Char(6)	No	V_packsku_qty.qty or 0
PACK_IND	Char(1)	Yes	Item_master.pack_ind
SIMPLE_PACK_IND	Char(1)	Yes	Item_master.simple_pack_ ind
ITEM_LEVEL	Integer(1)	Yes	Item_master.item_level
TRAN_LEVEL	Integer(1)	Yes	Item_master.tran_level
RETAIL_LABEL_TYPE	Char(6)	No	Item_master.retail_label_ type
CATCH_WEIGHT_IND	Char(1)	Yes	Item_master.catch_weight_ ind
SELLABLE_IND	Char(1)	Yes	Item_master.sellable_ind
ORDERABLE_IND	Char(1)	Yes	Item_master.orderable_ind
DEPOSIT_ITEM_TYPE	Char(6)	No	Item_master.deposit_item_ type
ITEM	Char(25)	Yes	Item_master.item

Integration Contract

Integration Type	Download from Merchandising
File Name	rmse_aip_purged_item.dat
Integration Contract	IntCon000136 rmse_aip_item_master.schema

The purged items output file is in fixed-length format matching to the schema definition in rmse_aip_purged_item.schema.

Output File Layout

Table 24–7 File Layout

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	Daily_purge.key_value

rmse_aip_store (Extract of Stores for AIP)

Module Name Rmse_aip_store.ksh

Description Extract of Stores for AIP

Functional Area Integration - AIP

Module Type Integration

Module Technology Ksh

Catalog ID RMS40

Wrapper Script rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts store information for integration with Oracle Retail Advanced Inventory Planning (AIP).

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–8 Key Tables Affected

Table	Select	Insert	Update	Delete
STORE	Yes	No	No	No
STORE_FORMAT	Yes	No	No	No
CODE_DETAIL	Yes	No	No	No

I/O Specification

Integration Type Download from Merchandising

File Name Determined by runtime parameter

Integration Contract IntCon000080

rmse_aip_store.schema

Output File Layout

Table 24–9 File Layout

Field Name	Field Type	Required	Description
STORE	Integer(11)	Yes	Store.store
STORE_NAME	Char(20)	Yes	Store.store_name
DISTRICT	Integer(11)	Yes	Store.district
STORE_CLOSE_DATE	Date	No	Store.store_close_date
STORE_OPEN_DATE	Date	Yes	Store.store_open_date
STORE_CLASS	Char(1)	Yes	Store.store_class
STORE_CLASS_ DESCRIPTION	Char(40)	Yes	Code_detail.code_desc
STORE_FORMAT	Integer(5)	No	Store.store_format
FORMAT_NAME	Char(20)	No	Store_format.format_name
STOCKHOLDING_IND	Char(1)	Yes	Store.stockholding_ind
REMERCH_IND	Char(1)	Yes	Store.remerch_ind
CLOSING_STORE_IND	Char(1)	Yes	'N' if Store.store_close_date is empty, else 'Y'

Design Assumptions

N/A

rmse_aip_wh (Extract of Warehouses for AIP)

Module Name	rmse_aip_wh.ksh
Description	Extract of Warehouses for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS35
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts warehouse information from Merchandising for integration with Oracle Retail Advanced Inventory Planning (AIP).

The script produces three extract files:

- rmse_aip_wh.dat
- rmse_aip_wh.txt

- rmse_aip_wh_type.txt

Only stock holding warehouses are extracted to the rmse_aip_wh.txt and rmse_aip_wh_type.txt files

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–10 Key Tables Affected

Table	Select	Insert	Update	Delete
WH	Yes	No	No	No

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_wh.dat
Integration Contract	IntCon000085 rmse_aip_wh_dat.schema

Output File Layout

Table 24–11 File Layout

Field Name	Field Type	Required	Description
WH	Integer(11)	Yes	Wh.wh
WH_NAME	Char(20)	Yes	Wh.wh_name
FORECAST_WH_IND	Char(1)	Yes	Wh.forecast_wh_ind
STOCKHOLDING_IND	Char(1)	Yes	Wh.stockholding_ind
WH_TYPE	Char(6)	No	Wh.vwh_type

I/O Specification

Integration	Download from Merchandising
File Name	rmse_aip_wh.txt
Integration Contract	IntCon000137 rmse_aip_wh_dat.schema

Output File Layout

Table 24–12 File Layout

Field Name	Field Type	Required	Description
WAREHOUSE_CHAMBER	Char(20)	Yes	Wh.wh

Table 24–12 (Cont.) File Layout

Field Name	Field Type	Required	Description
WAREHOUSE_CHAMBER_ DESCRIPTION	Char(40)	Yes	Wh.wh_name
WAREHOUSE	Integer(20)	Yes	Wh.wh
WAREHOUSE_ DESCRIPTION	Char(40)	Yes	Wh.wh_name

I/O Specification

Integration Type Download from Merchandising

File Name rmse_aip_wh_type.txt

Integration Contract IntCon000138

rmse_aip_wh_dat.schema

Output File Layout

Table 24–13 File Layout

Field Name	Field Type	Required	Description
WAREHOUSE	Integer(20)	Yes	Wh.wh
WH_TYPE	Char(6)	No	Wh.wh_type

Design Assumptions

N/A

rmse_aip_substitute_items (Extract of Substitute Items for AIP)

Module Name rmse_aip_substitute_item.ksh

Description Extract of Substitute Items for AIP

Functional Area Integration - AIP

Module Type Integration

Module Technology Ksh

Catalog ID RMS38

Wrapper Script rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts substitute item information from Merchandising for integration with Oracle Retail Advanced Inventory Planning (AIP).

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–14 Key Tables Affected

Table	Select	Insert	Update	Delete
SUB_ITEMS_DETAIL	Yes	No	No	No

I/O Specification

Integration Download from Merchandising

File Name rmse_aip_substitute_items.dat

Integration Contract IntCon000082

rmse_aip_substitute_items.schema

Output File Layout

Table 24–15 File Layout

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	Sub_items_detail.item
LOCATION	Integer(10)	Yes	Sub_items_detail.location
SUB_ITEM	Char(25)	Yes	Sub_items_detail.sub_item
LOC_TYPE	Char(1)	Yes	Sub_items_detail.loc_type
START_DATE	Date	No	Sub_items_detail.start_date
END_DATE	Date	No	Sub_items_detail.end_date
SUBSTITUTE_REASON	Char(1)	No	Sub_items_ detail.substitute_reason

Design Assumptions

N/A

rmse_aip_suppliers (Extract of Suppliers for AIP)

Module Name rmse_aip_suppliers.ksh

Description Extract of Suppliers for AIP

Functional Area Integration - AIP

Module Type Integration

Module Technology Ksh

Catalog ID RMS37

Wrapper Script rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts supplier/supplier site information for integration with Oracle Retail Advanced Inventory Planning (AIP).

The script produces three extract files:

- rmse_aip_suppliers.dat
- splr.txt
- dmx_dirspl.txt

Splr.txt and dmx_dirspl.txt only contain active suppliers (sups.sup_status = 'A').

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–16 Key Tables Affected

Table	Select	Insert	Update	Delete
SUPS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_suppliers.dat
Integration Contract	IntCon000083 rmse_aip_suppliers.schema

Output File Layout

Table 24–17 File Layout

Field Name	Field Type	Required	Description
SUPPLIER	Integer(11)	Yes	Sups.supplier
SUP_NAME	Char(32)	Yes	Sups.sup_name

I/O Specification

Integration Type	Download from Merchandising
File Name	splr.txt
Integration Contract	IntCon000175 rmse_aip_suppliers.schema

Output File Layout

Table 24–18 File Layout

Field Name	Field Type	Required	Description
SUPPLIER	Integer(20)	Yes	Sups.supplier
SUPPLIER_DESCRIPTION	Char(40)	Yes	Sups.sup_name

I/O Specification

Integration Type Download from Merchandising

File Name dmx_dirspl.txt

Integration Contract IntCon000176

rmse_aip_suppliers.schema

Output File Layout

Table 24–19 File Layout

Field Name	Field Type	Required	Description
SUPPLIER	Integer(20)	Yes	Sups.supplier
DIRECT_SUPPLIER	Char(1)	Yes	If sup.dsd_ind = 'Y' then 1, else if sup.dsd_ind = 'N' then 0

Design Assumptions

N/A

rmse_aip_alloc_in_well (Extract of Allocations in the Well Quantities for AIP)

Module Name rmse_aip_alloc_in_well.ksh

Description Extract of Allocations in the Well Quantities for AIP

Functional Area Integration - AIP

Module Type Integration

Module Technology Ksh

Catalog ID RMS20

Wrapper Script rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising "in the well" allocation quantities for integration with Oracle Retail Advanced Inventory Planning (AIP). In the well pertains to inventory that has been reserved by allocations in approved or reserved status. The expected release date is also included in the extract.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_alloc_in_well.dat
Integration Contract	IntCon000066 rmse_aip_alloc_in_well.schema

File Layout

Table 24–20 File Layout

Field Name	Field Type	Required	Description
DAY	Char(9)	Yes	alloc_header.release_date
LOC	Integer(20)	Yes	Alloc_header.wh
ITEM	Char(20)	Yes	Formal Case Type: If simple pack then and alloc_detail.to_loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item
ORDER_MULTIPLE	Char(6)	Yes	Formal Case Type: If simple pack and alloc_detail.to_loc_type = 'W' then this would be v_packsku_qty.qty of the pack component else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)

Table 24–20 (Cont.) File Layout

Field Name	Field Type	Required	Description
ALLOC_RESERVE_QTY	Char(8)	Yes	Formal Case Type: Alloc_detail.qty_allocated - alloc_detail.qty_received. Resulting quantity is multiplied by V_packsku_ qty.qty if item is a pack. Informal Case Type: Alloc_detail.qty_allocated - alloc_detail.qty_received expressed in multiples of the primary case size. The remainder is expressed in Standard UOM.
ORDER_NO	Integer(12)	No	Order number

The reject file rmse_aip_alloc_in_well_reject_ord_mult.txt is in pipe delimited (|) format

Table 24–21 File Layout

Field Name	Field Type	Required	Description
DAY	Char(9)	Yes	alloc_header.release_date
LOC	Integer(20)	Yes	Alloc_header.wh
ITEM	Char(20)	Yes	Formal Case Type: If simple pack then and alloc_detail.to_loc_type = 'S' then this would be the component of the pack in v_ packsku_qty else item_ master.item. Informal Case Type: Item_master.item
ORDER_MULTIPLE	Char(6)	Yes	Formal Case Type: If simple pack and alloc_ detail.to_loc_type = 'W' then this would be v_packsku_ qty.qty of the pack component else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_ pack_size and (ti * hi * supp_packsize)

Table 24–21 (Cont.) File Layout

Field Name	Field Type	Required	Description
ALLOC_RESERVE_QTY	Char(8)	Yes	Formal Case Type: Alloc_detail.qty_allocated - alloc_detail.qty_received. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Alloc_detail.qty_allocated - alloc_detail.qty_received expressed in multiples of the primary case size. The remainder is expressed in Standard UOM.
ORDER_NO	Integer(12)	No	Order number

rmse_aip_cl_po (Extract of AIP Generated POs, Allocations and Transfers Cancelled or Closed in Merchandising for AIP)

Module Name	rmse_aip_cl_po.ksh
Description	Extract of AIP Generated POs, Allocations and Transfers Cancelled or Closed in Merchandising for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS21
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts from Merchandising cancelled or closed purchase orders, transfers and allocations for integration with Oracle Retail Advanced Inventory Planning (AIP). Only records that meet the following criteria below are extracted:

For Purchase Orders:

- Close is not NULL
- Origin indicator is 6 (external system generated)
- Order close date is > Retl_extract_dates.last_extr_closed_pot_date

For Transfers:

- Transfer close date is not NULL
- Transfer is generated by AIP
- Order close date is > Retl_extract_dates.last_extr_closed_pot_date

For Allocations:

- Allocation close date is not NULL
- Allocation is generated by AIP
- Allocation close date > Retl_extract_dates.last_extr_closed_pot_date

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	output file closed_order.txt
Integration Contract	IntCon000068 rmse_aip_cl_po.schema

Output File Layout

Table 24–22 File Layout

File Name	Field Type	Required	Description
ORDER_NUMBER	Integer(12)	Yes	Ordhead.order_no or tsfhead.tsf_no or alloc_ header.alloc_no
ORDER_TYPE	Char(1)	Yes	'P' for purchase orders or 'T' for transfers or 'A' for allocations

rmse_aip_future_delivery_alloc (Extract of Allocation Quantities for Future Delivery for AIP)

Module Name	rmse_aip_future_delivery_alloc.ksh
Description	Extract of Allocation Quantities for Future Delivery for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS28
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising in-transit and on-order allocation quantities for future delivery for integration with AIP.

For warehouse-inbound transactions, the allocation number will be included as the transaction number in the output file. For store-inbound transactions, NULL will be included as the transaction number in the output file and transaction quantity will be rolled up by item/store/day. Both standalone allocations and cross-docked allocations from a PO will be extracted, but cross-docked allocations from a PO associated with a customer order will NOT be extracted.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_future_delivery_alloc.dat
Integration Contract	IntCon000069 rmse_aip_future_delivery_alloc.schema

Output File Layout

Table 24–23 File Layout

Field Name	Field Type	Required	Description
TRANSACTION_NUM	Integer(12)	No	If alloc_detail.to_loc_type ='W' then value will be Alloc_header.alloc_no else null
DAY	Char(9)	Yes	'D'|| Alloc_header.release_date + transit_times.transit_time
SUPPLIER	Integer(20)	No	If there is no associated order then primary supplier on item_supplier.supplier else ordhead.supplier
LOC	Integer(20)	Yes	Alloc_detail.to_loc
LOC_TYPE	Char(1)	Yes	Alloc_detail.to_loc_type
ITEM	Char(20)	Yes	Formal Case Type: If simple pack then and alloc_detail.to_loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item

Table 24–23 (Cont.) File Layout

Field Name	Field Type	Required	Description
ORDER_MULTIPLE	Char (6)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
IN_TRANSIT_ALLOC_QTY	Char (8)	Yes	Formal Case Type: Alloc_detail.Qty_transferred - Alloc_detail.Qty_received. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Alloc_detail.Qty_transferred - Alloc_detail.Qty_received expressed in the primary case size. Remainder is in Standard UOM
ON_ORDER_ALLOC_QTY	Char (8)	Yes	Formal Case Type: Alloc_detail.Qty_allocated - Alloc_detail.Qty_transferred. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Alloc_detail.Qty_allocated - Alloc_detail.Qty_transferred expressed in the primary case size. Remainder is in Standard UOM

The reject file rmse_aip_future_delivery_alloc_reject_ord_mult.txt is in pipe delimited (|) format.

Table 24–24 File Layout

Field Name	Field Type	Required	Description
TRANSACTION_NUM	Integer(12)	No	If alloc_detail.to_loc_type ='W' then value will be Alloc_header.alloc_no else null

Table 24–24 (Cont.) File Layout

Field Name	Field Type	Required	Description
DAY	Char(9)	Yes	'D'|| Alloc_header.release_date + transit_times.transit_time
SUPPLIER	Integer(20)	No	If there is no associated order then primary supplier on item_supplier.supplier else ordhead.supplier
LOC	Integer(20)	Yes	Alloc_detail.to_loc
LOC_TYPE	Char(1)	Yes	Alloc_detail.to_loc_type
ITEM	Char(20)	Yes	Formal Case Type: If simple pack then and alloc_detail.to_loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item
ORDER_MULTIPLE	Char (6)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
IN_TRANSIT_ALLOC_QTY	Char (8)	Yes	Formal Case Type: Alloc_detail.Qty_transferred - Alloc_detail.Qty_received. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Alloc_detail.Qty_transferred - Alloc_detail.Qty_received expressed in the primary case size. Remainder is in Standard UOM

Table 24–24 (Cont.) File Layout

Field Name	Field Type	Required	Description
ON_ORDER_ALLOC_QTY	Char (8)	Yes	Formal Case Type: Alloc_detail.Qty_allocated - Alloc_detail.Qty_ transferred. Resulting quantity is multiplied by V_ packsku_qty.qty if item is a pack. Informal Case Type: Alloc_detail.Qty_allocated - Alloc_detail.Qty_transferred expressed in the primary case size. Remainder is in Standard UOM

rmse_aip_future_delivery_order (Extract of Purchase Order Quantities for Future Delivery to AIP)

Module Name	rmse_aip_future_delivery_order.ksh
Description	Extract of Purchase Order Quantities for Future Delivery to AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS22
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising purchase order quantities for future delivery for integration with Oracle Retail Advanced Inventory Planning (AIP).

For warehouse-inbound transactions, the order number will be included as the transaction number in the output file. For store-inbound transactions, NULL will be included as the transaction number in the output file and transaction quantity will be rolled up by item/store/day. Both standalone POs and cross-docked POs to a transfer or allocation will be extracted, but POs associated with a customer order will NOT be extracted

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_future_delivery_order.dat
Integration Contract	IntCon000070 rmse_aip_future_delivery_order.schema

File Layout

Table 24–25 File Layout

Field Name	Field Type	Required	Description
TRANSACTION_NUM	Integer(12)	No	If ordloc.loc_type ='W' then value will be ordloc.order_no else null
DAY	Char(9)	Yes	'D' || Ordhead.not_after_date
SUPPLIER	Integer(20)	Yes	Ordhead.supplier
LOC	Integer(20)	Yes	Ordloc.location
ITEM	Char(20)	Yes	Formal Case Type: If simple pack and ordloc.loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item
ORDER_MULTIPLE	Char(6)	Yes	Formal Case Type: If ordloc.loc_type = 'S' then 1 If ordloc.loc_type = 'W' and (ordloc.qty_ordered - ordloc.qty_received) >= item_supp_country.supp_pack_size and a simple pack then V_packsku_qty.qty else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
PO_QTY	Char(8)	Yes	(Ordloc.qty_ordered - Ordloc.qty_received) or 0
LOC_TYPE	Char(1)	Yes	Ordloc.loc_type

The reject file rmse_aip_future_delivery_order_reject_ord_mult.txt is in pipe delimited (|) format.

Table 24–26 File Layout

Field Name	Field Type	Required	Description
TRANSACTION_NUM	Integer(12)	No	If ordloc.loc_type ='W' then value will be ordloc.order_no else null
DAY	Char(9)	Yes	'D' || Ordhead.not_after_date
SUPPLIER	Integer(20)	Yes	Ordhead.supplier
LOC	Integer(20)	Yes	Ordloc.location
ITEM	Char(20)	Yes	Formal Case Type: If simple pack and ordloc.loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item
ORDER_MULTIPLE	Char(6)	Yes	Formal Case Type: If ordloc.loc_type = 'S' then 1 If ordloc.loc_type = 'W' and (ordloc.qty_ordered - ordloc.qty_received) >= item_supp_country.supp_pack_size and a simple pack then V_packsku_qty.qty else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
PO_QTY	Char(8)	Yes	(Ordloc.qty_ordered - Ordloc.qty_received) or 0
LOC_TYPE	Char(1)	Yes	Ordloc.loc_type

rmse_aip_future_delivery_tsf (Extract On Order and In Transit Transfer Quantities for Future Delivery for AIP)

Module Name	rmse_aip_future_delivery_tsf.ksh
Description	Extract On Order and In Transit Transfer Quantities for Future Delivery for AIP
Functional Area	Integration - AIP
Module Type	Integration

Module Technology	Ksh
Catalog ID	RMS29
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising on-order and in-transit transfer quantities for future delivery for Integration with AIP.

For warehouse-inbound transactions, the transfer number will be included as the transaction number in the output file. For store-inbound transactions, NULL will be included as the transaction number in the output file and transaction quantity will be rolled up by item/store/day.

Transfers created by RMS's franchise ordering/returning processes will not be extracted.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Integration

Integration Type	Download from Merchandising
File Name	rmse_aip_future_delivery_tsf.dat
Integration Contract	IntCon000071 rmse_aip_future_delivery_tsf.schema

Output File Layout

Table 24–27 File Layout

Field Name	Field Type	Required	Description
TRANSACTION_NUM	Integer(12)	No	If tsfhead.to_loc_type ='W' then value will be tsfhead.tsf_no else null
DAY	Char(9)	Yes	'D' || tsfhead.delivery_date + transit_times.transit_time
SUPPLIER	Integer(20)	No	Item_supp_country.supplier
LOC	Integer(20)	Yes	Shipitem_inv_flow.to_loc if tsfhead.to_loc_type = 'W' and tsfhead.tsf_type = 'EG else Tsfhead.to_loc

Table 24–27 (Cont.) File Layout

Field Name	Field Type	Required	Description
ITEM	Char(20)	Yes	Formal Case Type: If simple pack and tsfhead.to_loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item
ORDER_MULTIPLE	Char (6)	Yes	Formal Case Type: If simple pack and tsfhead.to_loc_type = 'W' the v_packsku_qty.qty else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
TSF_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.tsf_qty - tsfdetail.received_qty. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Tsfdetail.tsf_qty - tsfdetail.received_qty expressed in the primary case size. Remainder is in Standard UOM
IN_TRANSIT_TSF_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.ship_qty - tsfdetail.received_qty. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Tsfdetail.ship_qty - tsfdetail.received_qty expressed in the primary case size. Remainder is in Standard UOM

Table 24–27 (Cont.) File Layout

Field Name	Field Type	Required	Description
ON_ORDER_TSF_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty expressed in the primary case size. Remainder is in Standard UOM
LOC_TYPE	Char(1)	Yes	Tsfhead.to_loc_type
TSF_TYPE	Char(6)	Yes	Tsfhead.tsf_type

The reject file rmse_aip_future_delivery_tsf_reject_ord_mult.txt is in pipe delimited (|) format.

Table 24–28 File Layout

Field Name	Field Type	Required	Description
TRANSACTION_NUM	Integer(12)	No	If tsfhead.to_loc_type ='W' then value will be tsfhead.tsf_no else null
DAY	Char(9)	Yes	'D' || tsfhead.delivery_date + transit_times.transit_time
SUPPLIER	Integer(20)	No	Item_supp_country.supplier
LOC	Integer(20)	Yes	Shipitem_inv_flow.to_loc if tsfhead.to_loc_type = 'W' and tsfhead.tsf_type = 'EG else Tsfhead.to_loc
ITEM	Char(20)	Yes	Formal Case Type: If simple pack and tsfhead.to_loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item

Table 24–28 (Cont.) File Layout

Field Name	Field Type	Required	Description
ORDER_MULTIPLE	Char (6)	Yes	Formal Case Type: If simple pack and tsfhead.to_loc_type = 'W' the v_packsku_qty.qty else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_ pack_size and (ti * hi * supp_packsize)
TSF_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.tsf_qty - tsfdetail.received_qty. Resulting quantity is multiplied by V_packsku_ qty.qty if item is a pack. Informal Case Type: Tsfdetail.tsf_qty - tsfdetail.received_qty expressed in the primary case size. Remainder is in Standard UOM
IN_TRANSIT_TSF_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.ship_qty - tsfdetail.received_qty. Resulting quantity is multiplied by V_packsku_ qty.qty if item is a pack. Informal Case Type: Tsfdetail.ship_qty - tsfdetail.received_qty expressed in the primary case size. Remainder is in Standard UOM
ON_ORDER_TSF_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty. Resulting quantity is multiplied by V_ packsku_qty.qty if item is a pack. Informal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty expressed in the primary case size. Remainder is in Standard UOM
LOC_TYPE	Char(1)	Yes	Tsfhead.to_loc_type
TSF_TYPE	Char(6)	Yes	Tsfhead.tsf_type

rmse_aip_item_loc_traits (Extract of Shelf Life on Receipt Location Trait for AIP)

Module Name	rmse_aip_item_loc_traits.ksh
Description	Extract of Shelf Life on Receipt Location Trait for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS23
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts from Merchandising item location traits information for integration with Oracle Retail Advanced Inventory Planning (AIP). Only the following items are extracted:

- Approved, non-pack and forecastable
- Approved and a simple pack item whose component is forecastable.
- Items which are intentionally ranged to the location.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_item_loc_traits.dat
Integration Contract	IntCon000072 rmse_aip_item_loc_traits.schema

Output File Layout

Table 24–29 File Layout

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	Item_master.item
LOC	Integer(10)	Yes	Item_loc_traits.loc
REQ_SHELF_LIFE_ ON_RECEIPT	Integer(8)	No	Item_loc_traits.req_shelf_ life_on_ receipt

rmse_aip_item_retail (Extract of Forecasted Items for AIP)

Module Name	rmse_aip_item_retail.ksh
Description	Extract of Forecasted Items for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS24
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts from Merchandising item information required by the item transformation script aipt_item.ksh for integration with Oracle Retail Advanced Inventory Planning (AIP).

Records that meet the following criteria are extracted:

Non-Pack Items

- Approved and transaction level items
- Have supplier pack sizes greater than 1
- Forecastable
- Inventory items

Simple Pack Components

- Component of approved and transaction level simple packs
- Components are forecastable
- Simple packs are inventory items

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_item_retail.dat
Integration Contract	IntCon000074 rmse_aip_item_retail.schema

Output File Layout

Table 24–30 File Layout

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	Item_master.item
AIP_SKU	Char(25)	Yes	Item_master.item
SUBCLASS	Integer(5)	Yes	Item_master.subclass
CLASS	Integer(5)	Yes	Item_master.class
DEPT	Integer(5)	Yes	Item_master.dept
STANDARD_UOM	Char(4)	Yes	Item_master.standard_uom
STANDARD_UOM_ DESCRIPTION	Char(20)	Yes	Uom_class.uom_desc_ standard
SKU_TYPE	Char(6)	No	Non-pack items Item_master.handling_ temp. "0" if NULL. Simple pack components Item_master.handling_temp or NULL.
SKU_TYPE_ DESCRIPTION	Char(40)	No	Non-pack items Code_detail.code_desc . "0" if NULL. Simple pack components Code_detail.code_desc or NULL.
ORDER_MULTIPLE	Char(6)	Yes	1
PACK_QUANTITY	Char(6)	No	0

rmse_aip_item_sale (Extract of Scheduled Item Maintenance On/Off Sale Information for AIP)

Module Name	rmse_aip_item_sale.ksh
Description	Extract of Scheduled Item Maintenance On/Off Sale Information for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS31
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts on/off sale information for integration with Oracle Retail Advanced Inventory Planning (AIP). This integration is designed to be used in conjunction with Scheduled Item Maintenance functionality in Merchandising.

The script produces two output files, one containing on sale records and the other off sale records.

If a client does not use Scheduled Item Maintenance functionality to manage the on and off sale attributes of items at locations, the client does not need to run this program. Instead, the customer should create on/off sales information for AIP through a custom process.

This information extracted for AIP includes the status, status update date and order multiple for an item/location.

A status of 'A' indicates that an item/location is valid and can be ordered and sold. A status of 'C' indicates than an item/location is invalid and cannot be ordered or sold. The script only extracts items that meet the following criteria:

- In active status
- Transaction level
- Either non-pack or a simple pack
- Status in the SIT_DETAIL table is either 'A' or 'C'
- Status update date in the SIT_DETAIL table is greater than the current date

Only the order multiple for the primary supplier and primary supplier country is extracted.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	dm0_onseffdt_.txt
Integration Contract	IntCon000075 rmse_aip_item_on_sale.schema

Output File Layout

Table 24–31 File Layout

Field Name	Field Type	Required	Description
STORE	Integer(20)	Yes	Sit_explode.location
RMS_SKU	Char(20)	Yes	Sit_explode.item
ORDER_MULTIPLE	Char(6)	Yes	If item_master.pack_ind = 'Y' then v_packsku_qty.qty (for the component item) else item_supp_ country.order_multiple

Table 24–31 (Cont.) File Layout

Field Name	Field Type	Required	Description
ON_SALE_EFFECTIVE_ DATE	Date	Yes	Sit_detail.status_update_ date

Integration Contract

Integration Type	Download from Merchandising
File Name	dm0_ofseffdt_.txt
Integration Contract	IntCon000135 rmse_aip_item_off_sale.schema

Output File Layout

Table 24–32 File Layout

Field Name	Field Type	Required	Description
STORE	Integer(20)	Yes	Sit_explode.location
RMS_SKU	Char(20)	Yes	Sit_explode.item
ORDER_MULTIPLE	Char(6)	Yes	If item_master.pack_ind = 'Y' then v_packsku_qty.qty (for the component item) else item_supp_ country.order_multiple
OFF_SALE_EFFECTIVE_ DATE	Date	Yes	Sit_detail.status_update_ date

The reject file rmse_aip_item_sale_reject_ord_mult.txt is in pipe delimited (|) format.

File Layout

Table 24–33 File Layout

Field Name	Field Type	Required	Description
STORE	Integer(20)	Yes	Sit_explode.location
RMS_SKU	Char(20)	Yes	Sit_explode.item
ORDER_MULTIPLE	Char(6)	Yes	If item_master.pack_ind = 'Y' then v_packsku_qty.qty (for the component item) else item_supp_ country.order_multiple
OFF_SALE_EFFECTIVE_ DATE	Date	Yes	Sit_detail.status_update_ date
ON_SALE_EFFECTIVE_ DATE			

rmse_aip_item_supp_country (Extract of Order Multiples by Item/Supplier/Origin Country for AIP)

Module Name	rmse_aip_item_supp_country.ksh
Description	Extract of Order Multiples by Item/Supplier/Origin Country for AIP
Functional Area	Merchandising to AIP Integration
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS25
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising item-supplier information for integration with Oracle Retail Advanced Inventory Planning (AIP).

Three output files are produced by this extract. Two contain item-supplier information. The other is a reject file containing item suppliers with rejected order multiples.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_item_supp_country.dat
Integration Contract	IntCon000076 rmse_aip_item_supp_country.schema

File Layout

Table 24–34 File Layout

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	Item_supp_country.item
SUPPLIER	Integer(11)	Yes	Item_supp_country.supplier

Table 24–34 (Cont.) File Layout

Field Name	Field Type	Required	Description
ORDER_MULTIPLE	Integer(4)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
PRIMARY_SUPP_IND	Char(1)	Yes	Item_supp_country.primary_supp_ind

Integration Contract

Integration Type	Download from Merchandising
File Name	aip_dmx_prdsplks.txt
Integration Contract	IntCon000133 rmse_aip_dmx_prdspllks.schema

File Layout

Table 24–35 File Layout

Field Name	Field Type	Required	Description
SUPPLIER	Integer(20)	Yes	Item_supp_country.supplier
RMS_SKU	Char(20)	Yes	Item_supp_country.item
ORDER_MULTIPLE	Char (6)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
COMMODITY_SUPPLIER_LINKS	Char(1)	Yes	1

The reject file rmse_aip_item_supp_country_reject_ord_mult.txt is in pipe delimited (|) format.

File Layout

Table 24–36 File Layout

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	Item_supp_country.item
SUPPLIER	Integer(11)	Yes	Item_supp_country.supplier
ORDER_MULTIPLE	Char(6)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
PRIMARY_SUPP_IND	Char(1)	Yes	Item_supp_country.primary_supp_ind

rmse_aip_rec_qty (Extract of Received PO, Allocation and Transfer Quantities for AIP)

Module Name	rmse_aip_rec_qty.ksh
Description	Extract of Received PO, Allocation and Transfer Quantities for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS33
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts received PO, transfer and allocation quantities from Merchandising for integration with Oracle Retail Advanced Inventory Planning (AIP). Only records that meet the following criteria below are extracted:

For Purchase Orders:

- Close date on order header is NULL or close date is greater than or equal to the current date minus the maximum not after days
- Not after date on order header is not NULL
- Origin indicator is 6 (external system generated) on order header
- Received quantity is not NULL

For Transfers:

- Close date is NULL on transfer header or close date is greater than or equal to the current date minus the maximum not after days
- Transfer type on transfer header is 'AIP' (generated by AIP)
- Delivery date on transfer header is not NULL
- Received quantity is not NULL

For Allocations:

- Close date on allocation header is NULL or close date is greater than or equal to the current date minus the maximum not after days
- Origin indicator is 'AIP' on allocation header (generated by AIP)
- Release date on allocation header is not NULL
- Quantity received is not NULL
- Order number on allocation header is not NULL(AIP generated allocations will always have an order associated with them)

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–37 Key Tables Affected

Table	Select	Insert	Update	Delete
ORDHEAD	Yes	No	No	No
ORDLOC	Yes	No	No	No
ORDSKU	Yes	No	No	No
TSFHEAD	Yes	No	No	No
TSFDETAIL	Yes	No	No	No
V_PACKSKU_QTY	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No

I/O Specification

Integration Type	Download from Merchandising
File Name	received_qty.txt
Integration Contract	IntCon000079 rmse_aip_rec_qty.schema

Output File Layout

Table 24–38 File Layout

Field Name	Field Type	Required	Description
ORDER_NUMBER	Integer(12)	Yes	Ordhead.order_no or tsfhead.tsf_no or alloc_ header.alloc_no
ORDER_TYPE	Char(1)	Yes	'P' for purchase orders or 'T' for transfers or 'A' for allocations
RMS_SKU	Char(25)	Yes	Ordsku.item or tsfdetail.item or alloc_ header.item
ORDER_MULTIPLE	Char(6)	Yes	Ordsku.supp_pack_size or tsfdetail.supp_pack_size
PACK_QTY	Char(6)	Yes	If pack item then sum of V_ packsku_qty.qty else 0
STORE	Integer(10)	No	If ordloc.loc_type = 'S' then ordloc.location or If tsfhead.to_loc_type = 'S' then tsfhead.to_loc or If alloc_detail.to_loc_type = 'S' then alloc_detail.to_loc
WAREHOUSE	Integer(10)	No	If ordloc.loc_type = 'W' then ordloc.location or If tsfhead.to_loc_type = 'W' then tsfhead.to_loc or If alloc_detail.to_loc_type = 'W' then alloc_detail.to_loc
RECEIVED_DATE	Date	Yes	Ordhead.not_after_date or tsfhead.delivery_date or alloc_header.release_date
QUANTITY	Char(8)	No	Ordloc.qty_received or tsfdetail.received_qty or alloc_detail.qty_received

Design Assumptions

N/A

rmse_aip_store_cur_inventory (Extract of Store Current Inventory data for AIP)

Module Name	rmse_aip_store_cur_inventory.ksh
Description	Extract of Store Current Inventory data for AIP
Functional Area	Integration - AIP

Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS39
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising current inventory for store locations for integration with Oracle Retail Advanced Inventory Planning (AIP). This script requires an 'F' or 'D' parameter:

- F - full extract of items/locations. Multiple output files. One file per item_loc_soh partition.
- D - delta extract of items/locations for the current day's transactions as well as for the locations for which backorder message was received. Single output file.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–39 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_MASTER	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
STORE	Yes	No	No	No
IF_TRAN_DATA	Yes	No	No	No
IF_TRAN_DATA_TEMP	Yes	Yes	No	No
INV_RESV_UPDATE_TEMP	Yes	No	Yes	Yes
PACKITEM	Yes	No	No	No
DBA_TAB_PARTITIONS	Yes	No	No	No

I/O Specification

Integration Type	Download from Merchandising
File Name	sr0_curinv_{THREAD_NO}.txt
Integration Contract	IntCon000081 rmse_aip_store_cur_inventory.schema

Output File Layout

Table 24–40 File Layout

Field Name	Field Type	Required	Description
STORE	Integer(20)	Yes	Item_loc_soh.loc
RMS_SKU	Char(20)	Yes	Item_master.item
STORE_CUR_INV	Char (8)	No	Item_loc_soh.stock_on_hand - (item_loc_soh.tsf_reserved_qty + item_loc_soh.rtv_qty + item_loc_soh.non_sellable_qty + item_loc_soh.customer_resv)
BACKORDER_QUANTITY	Char (8)	No	Item_loc_soh.customer_backorder

Design Assumptions

N/A

rmse_aip_tsf_in_well (Extract of Transfer in the Well Quantities to AIP)

Module Name	rmse_aip_tsf _in_well.ksh
Description	Extract of Transfer in the Well Quantities to AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	Ksh
Catalog ID	RMS36
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts Merchandising "in the well" transfer quantities for integration with AIP. In the well pertains to inventory that has been reserved by an approved or shipped transfer. The expected delivery date is also included in the extract.

Transfers created by the Merchandising franchise ordering and return processes will not be extracted.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–41 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_MASTER	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_SUPPLIER	Yes	No	No	No
TSFHEAD	Yes	No	No	No
TSFDETAIL	Yes	No	No	No
SHIPITEM_INV_FLOW	Yes	No	No	No
TRANSIT_TIMES	Yes	No	No	No
V_WH	Yes	No	No	No
V_PACKSKU_QTY	Yes	No	No	No
PACKITEM	Yes	No	No	No

I/O Specification

Integration Type	Download from Merchandising
File Name	rmse_aip_tsf_in_well.dat
Integration Contract	IntCon000084 rmse_aip_tsf_in_well.schema

Output File Layout

Table 24–42 File Layout

Field Name	Field Type	Required	Description
DAY	Char(9)	Yes	tsfhead.delivery_date - transit_times.transit_time
LOC	Integer(20)	Yes	f tsfhead.from_loc type = 'W' and (tsfhead.tsf_type = 'EG' or tsfhead.tsf_type = 'CO' and OMS_IND = 'Y') then shipitem_inv_ flow.from_loc else tsfhead.from_loc
ITEM	Char(20)	Yes	Formal Case Type: If simple pack then and tsfhead.to_loc_type = 'S' then this would be the component of the pack in v_ packsku_qty else item_ master.item. Informal Case Type: Item_master.item

Table 24–42 (Cont.) File Layout

Field Name	Field Type	Required	Description
ORDER_MULTIPLE	Char (6)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
TSF_RESERVED_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty expressed in the primary case size. Remainder is in Standard UOM

The reject file rmse_aip_tsf_in_well_reject_ord_mult.txt is in pipe delimited (|) format.

Table 24–43 File Layout

Field Name	Field Type	Required	Description
DAY	Char(9)	Yes	tsfhead.delivery_date - transit_times.transit_time
LOC	Integer(20)	Yes	If tsfhead.from_loc type = 'W' and (tsfhead.tsf_type = 'EG' or tsfhead.tsf_type = 'CO' and OMS_IND = 'Y') then shipitem_inv_flow.from_loc else tsfhead.from_loc
ITEM	Char(20)	Yes	Formal Case Type: If simple pack then and tsfhead.to_loc_type = 'S' then this would be the component of the pack in v_packsku_qty else item_master.item. Informal Case Type: Item_master.item

Table 24–43 (Cont.) File Layout

Field Name	Field Type	Required	Description
ORDER_MULTIPLE	Char (6)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
TSF_RESERVED_QTY	Char (8)	Yes	Formal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty. Resulting quantity is multiplied by V_packsku_qty.qty if item is a pack. Informal Case Type: Tsfdetail.tsf_qty - tsfdetail.ship_qty expressed in the primary case size. Remainder is in Standard UOM

Design Assumptions

N/A

rmse_aip_wh_cur_inventory (Extract of Warehouse Current Inventory for AIP)

Module Name	rmse_aip_wh_cur_inventory.ksh
Description	Extract of Warehouse Current Inventory for AIP
Functional Area	Integration - AIP
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS34
Wrapper Script	rmswrap_aip.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script extracts current warehouse inventory information from Merchandising for integration with Oracle Retail Advanced Inventory Planning (AIP).

This script requires an 'F' or 'D' parameter:

- F - full extract of items/locations. Creates multiple files per warehouse. Files are concatenated into a single file upon successful completion.
- D - delta extract of items/locations for the current day's transactions as well as for the locations for which backorder message was received. Creates a single extract file.

The script creates a backup of the previous day's data file labeled with the date on which they were created.

Restart/Recovery

This is a standard Oracle Retail RETL script. No restart/recovery is used.

Key Tables Affected

Table 24–44 Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_MASTER	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No
ITEM_LOC_SOH	Yes	No	No	No
WH	Yes	No	No	No
ALLOC_DETAIL	Yes	No	No	No
ALLOC_HEADER	Yes	No	No	No
ORDHEAD	Yes	No	No	No
ITEM_SUPPLIER	Yes	No	No	No
PACKITEM	Yes	No	No	No
V_PACKSKU_QTY	Yes	No	No	No
IF_TRAN_DATA_TEMP	Yes	No	No	No

I/O Specification

Integration Type	Download from Merchandising
File Name	wr1_curinv.txt
Integration Contract	IntCon000092 rmse_aip_wh_cur_inventory.schema

Output File Layout

Table 24–45 File Layout

Field Name	Field Type	Required	Description
WAREHOUSE	Integer(20)	Yes	Item_loc_soh.loc
RMS_SKU	Char(20)	Yes	Item_master.item

Table 24–45 (Cont.) File Layout

Field Name	Field Type	Required	Description
ORDER_MULT	Char (6)	Yes	Formal Case Type: V_packsku_qty.qty for simple pack, else 1 Informal Case Type: One unique record for each item/supplier with order multiples of: 1, supp_pack_size, inner_pack_size and (ti * hi * supp_packsize)
WH_CUR_INV	Char (8)	Yes	Formal Case Type: ((Item_loc_soh.stock_on_hand - (item_loc_soh.tsf_reserved_qty + item_loc_soh.rtv_qty + item_loc_soh.non_sellable_qty + item_loc_soh.customer_resv)) - alloc_detail.qty_distro * Informal Case Type: ((Item_loc_soh.stock_on_hand - (item_loc_soh.tsf_reserved_qty + item_loc_soh.rtv_qty + item_loc_soh.non_sellable_qty + item_loc_soh.customer_resv)) - alloc_detail.qty_distro)
WH_BO_INV	Char (8)	Yes	Item_loc_soh.customer_backorder

Design Assumptions

N/A

25

Integration with General Ledger

Merchandising stages GL data for subsequent upload into a financial system. A set of batch processes gather and organize the data before using it to populate the staging table, STG_FIF_GL_DATA.

For more information about how data moves from these staging tables to the General Ledger of a financial application and other integration between Merchandising and financial applications, see *Oracle Retail Financial Integration for Oracle Retail Merchandise Operations Management and Oracle E-Business Suite Financials Implementation Guide*

Batch Design Summary

The following batch designs are included in this functional area:

- dealfinc.pc - Calculation & Interface of Fixed Deal Income for General Ledger
- fifgldn1.pc - Interface to General Ledger of Item/Loc Level Transactions
- fifgldn2. pc - Interface to General Ledger of Rolled Up Transactions
- fifgldn3. pc - Interface to General Ledger of Month Level Information
- gl_extract.ksh (Extraction of General Ledger transaction data from Merchandising and RESA)
- BDI_RFI_FinGenLdgr_Tx_PF_From_RMS_JOB (Finance General Ledger to RFI)

dealfinc (Calculation of Fixed Deal Income for General Ledger)

Module Name	dealfinc.pc
Description	Calculation & Interface of Fixed Deal Income for General Ledger
Functional Area	Integration - General Ledger
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS65
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module writes to the STG_FIF_GL_DATA financial staging table to perform stock ledger processing for fixed deals. It splits deal income over all dept/class/subclass locations on the deal. This prorated income is written to the general ledger under a suitable cost center mapping.

Restart/Recovery

The logical unit of work for this program is a DEAL_ID. The database commit takes place when number of deal records processed is equal to the commit max counter in the restart control table.

I/O Specification

Integration Type	Download from Merchandising
File Name	N/A
Integration Contract	IntCon000019 STG_FIF_GL_DATA table

Design Assumptions

N/A

fifgldn1 (Interface to General Ledger of Item/Loc Level Transactions)

Module Name	fifgldn1.pc
Description	Interface to General Ledger of Item/Loc Level Transactions
Functional Area	General Ledger
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS66
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program extracts the detailed stock ledger information for certain transaction types on a daily basis in order to bridge the information to an interfaced financial application. The program reads from the IF_TRAN_DATA table for each transaction type/amount type and posts it to the Oracle Retail General Ledger staging table at the SKU detail level.

Restart/Recovery

The logical unit of work is department/class/subclass. The batch is multithreaded using the v_restart_dept view.

I/O Specification

Integration Type	Download from Merchandising
File Name	N/A
Integration Contract	IntCon000019 STG_FIF_GL_DATA table

Design Assumptions

N/A

fifgldn2 (Interface to General Ledger of Rolled Up Transactions)

Module Name	fifgldn2.pc
Description	Interface to General Ledger of Rolled Up Transactions
Functional Area	Integration - General Ledger
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS67
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program summarizes stock ledger data from the transaction staging table (IF_TRAN_DATA) based on the level of information required and writes it to the financial general ledger staging table. The transactions extracted are determined by the CODE_TYPE 'GLRT' (General Ledger Rolled Transactions). The written information can then be extracted by the financial applications. Stock ledger information may be rolled-up at department, class or subclass level. The level at which information is rolled-up to is determined by the system parameter GL_ROLLUP.

Restart/Recovery

The logical unit of work is dependent on the level of rollup defined in system_options.gl_rollup. It can be department (department rollup), department/class (class rollup) or department/class/subclass (subclass rollup). The batch is multithreaded using the v_restart_dept view.

I/O Specification

Integration Type	Download from Merchandising
File Name	N/A
Integration Contract	IntCon000019 STG_FIF_GL_DATA table

Design Assumptions

N/A

fifgldn3 (Interface to General Ledger of Month Level Information)

Module Name	fifgldn3.pc
Description	General Ledger Interface 3
Functional Area	Interface to General Ledger of Month Level Information
Module Type	Integration
Module Technology	ProC
Catalog ID	RMS68
Wrapper Script	rmswrap_multi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program summarizes stock ledger data from the monthly stock ledger table based on the level of information required and writes it to the financial general ledger staging table. The transactions extracted are determined by the CODE_TYPE 'GLRT' (general ledger rolled transactions). Written information is then sent to the financial application. Stock ledger information may be rolled-up at department, class or subclass level. The level at which information is rolled-up to is determined by the system parameter GL_ROLLUP.

Restart/Recovery

The logical unit of work is dependent on the level of rollup defined in system_ options.gl_rollup. It can be department (department rollup), department/class (class rollup) or department/class/subclass (subclass rollup). The batch is multithreaded using the restart all locations view.

I/O Specification

Integration Type	Download from Merchandising
File Name	N/A

Integration Contract	IntCon000019 STG_FIF_GL_DATA table

Design Assumptions

N/A

gl_extract.ksh (Extraction of General Ledger transaction data from Merchandising and Sales Audit)

Module Name	gl_extract.ksh
Description	Extraction of General Ledger transaction data from Merchandising and Sales Audit to be interfaced to third party GL/Financial system
Functional Area	Integration to General Ledger
Module Type	Integration
Module Technology	ksh
Catalog ID	RMS495
Wrapper Script	rmswrap_shell_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch job will extract general ledger transaction data from Sales Audit and Merchandising into a file. Data to be extracted will be pulled off from the STG_FIF_GL_DATA table. Once the data is extracted into the file batch will purge the data from the table.

Restart/Recovery

N/A

I/O Specification

Integration Type	Extract from Merchandising
File Name	GL_EXTRACT_[#date].dat
Integration Contract	Na

Output File Layout

The output file is comma delimited with the following fields:

Record Name	Field Name
All records have the same structure	SET_OF_BOOKS_ID
	ACCOUNTING_DATE
	CURRENCY_CODE
	STATUS
	DATE_CREATED
	CREATED_BY
	ACTUAL_FLAG
	USER_JE_CATEGORY_NAME
	USER_JE_SOURCE_NAME
	CURRENCY_CONVERSION_DATE
	CURRENCY_CONVERSION_TYPE
	ACCT_SEGMENT1
	ACCT_SEGMENT2
	ACCT_SEGMENT3
	ACCT_SEGMENT4
	ACCT_SEGMENT5
	ACCT_SEGMENT6
	ACCT_SEGMENT7
	ACCT_SEGMENT8
	ACCT_SEGMENT9
	ACCT_SEGMENT10
	ENTERED_DR_AMOUNT
	ENTERED_CR_AMOUNT
	TRANSACTION_DATE
	REFERENCE1
	REFERENCE2
	REFERENCE3
	REFERENCE4
	REFERENCE5
	ATTRIBUTE1
	ATTRIBUTE2
	ATTRIBUTE3
	ATTRIBUTE4
	ATTRIBUTE5
	ATTRIBUTE6
	PERIOD_NAME
	CODE_COMBINATION_ID

Record Name	Field Name
	PGM_NAME
	ACCT_SEGMENT11
	ACCT_SEGMENT12
	ACCT_SEGMENT13
	ACCT_SEGMENT14
	ACCT_SEGMENT15
	ACCT_SEGMENT16
	ACCT_SEGMENT17
	ACCT_SEGMENT18
	ACCT_SEGMENT19
	ACCT_SEGMENT20
	REFERENCE_TRACE_ID
	PRIM_CURRENCY_CODE
	PRIM_ENTERED_DR_AMOUNT
	PRIM_ENTERED_CR_AMOUNT
	FIN_GL_SEQ_ID
	PROCESSED_FLAG

Design Assumptions

N/A

Finance General Ledger to RFI (BDI_RFI_FinGenLdgr_Tx_PF_From_RMS_JOB)

Module Name	BDI_RFI_FinGenLdgr_Tx_PF_From_RMS_JOB
Description	Extracts financial general ledger to RFI
Functional Area	Finance
Module Type	Integration
Module Technology	BDI Job
Catalog ID	N/A
Runtime Parameters	FinGenLdgr_Tx_ProcessFlow_From_RMS FinGenLdgr_Tx_Extractor

Design Overview

Merchandising extracts financial general ledger to RFI on a daily basis. It utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement from Merchandising to RFI.

The batch job BDI_RFI_FinGenLdgr_Tx_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RFI_FinGenLdgr_Tx_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Kicks off BDI process FinGenLdgr_Tx_
ProcessFlow_From_RMS"/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value=""/>
                <property name="predicateFunction" value=""/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RFI_FinGenLdgr_Tx_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It invokes a BDI process flow (FinGenLdgr_Tx_ProcessFLow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to the target application RFI:

- Extractor job (FinGenLdgr_Tx_ExtractorJob) calls BDI_FINANCIAL_SQL.FIF_GL_DATA_UP function to extract data from Merchandising table STG_FIF_GL_DATA to BDI outbound staging table FIF_GL_DATA_OUT.
- BDI will transport and import the data in FIF_GL_DATA_OUT to RFI.

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Daily
Scheduling Consideration	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
STG_FIF_GL_DATA	Yes	No	No	No
FIF_GL_DATA_OUT	Yes	Yes	No	No
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No

Table	Select	Insert	Update	Delete
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

Refer to FinGenLdgr_Tx_BdiInterfaceModule.xml.

26

Integration with Oracle Retail Planning and Forecasting

Merchandising provides critical foundation and transactional information to the Oracle Retail planning and forecasting solutions. Because the planning and forecasting solutions are built on the Retail Predictive Application Server (RPAS), several of the integrations from Merchandising are used by more than one of the planning and forecasting solutions. Additionally, Merchandising receives data back from some of these solutions. The table below summarizes the key integration points by solution.

Integration to Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)

Description	Program
Calendar Extract to Planning and Forecasting	BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB
Currency Rates Extract to Planning and Forecasting	BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB
Merchandise Hierarchy and Item Extract to Planning and Forecasting	BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB
Organization Hierarchy Extract to Planning and Forecasting	BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB
Store Extract to Planning and Forecasting	BDI_RPAS_Store_Fnd_PF_From_RMS_JOB
Inventory Extract to Planning	BDI_MFP_Inventory_Tx_PF_From_RMS_JOB
On Order Extract to Planning	BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB
Transaction Data Extract to Planning	BDI_MFP_TranData_Tx_PF_From_RMS_JOB

Integration to Oracle Retail Assortment and Item Planning for Fashion/ Softlines Cloud Service (APCS)

Description	Program
Brand Extract to Planning	BDI_RPAS_Brand_Fnd_PF_From_RMS_JOB

Description	Program
Calendar Extract to Planning and Forecasting	BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB
Currency Rates Extract to Planning and Forecasting	BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB
Differentiator Extract to Planning	BDI_RPAS_Diff_Fnd_PF_From_RMS_JOB
Merchandise Hierarchy and Item Extract to Planning and Forecasting	BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB
Organization Hierarchy Extract to Planning and Forecasting	BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB
Store Extract to Planning and Forecasting	BDI_RPAS_Store_Fnd_PF_From_RMS_JOB
Supplier Extract to Planning	BDI_RPAS_Supplier_Fnd_PF_From_RMS_JOB
UDA Extract to Planning	BDI_RPAS_UdaAndUdaValues_Fnd_PF_From_RMS_JOB
UDA Item Extract to Planning and Forecasting	BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB
Inventory Extract to Planning	BDI_MFP_Inventory_Tx_PF_From_RMS_JOB
On Order Extract to Planning	BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB
Transaction Data Extract to Planning	BDI_MFP_TranData_Tx_PF_From_RMS_JOB

Integration from Oracle Retail Assortment and Item Planning for Fashion/ Softlines Cloud Service (APCS)

In APCS, placeholder items can be defined as part of the planning process and, once finalized, sent to Merchandising through the Item Induction process, which allows an item to be uploaded for completion of setup and approval so that it can be used in ordering and other processes. This batch process works based on pre-defined templates that indicate what Merchandising should expect from the sending solution. For the APCS integration, the template that is used is Style and Style/Color Template (STYLE_COLOR_ITEM_DATA), intended to support fashion items only. It includes the data elements that APCS will send if a new style or style/color is created. This template contains three data key data elements for new items:

- **Item Master** - At the item level, APCS will send the department, class, subclass, item ID, description, item level, brand, and diff 1 information (for the style/color level only). The template will also default the following item attributes:
  - Catch weight, diff aggregate flags 2-4, forecast, pack, and primary ref item flags will be defaulted to N
  - Inventory, merchandise, orderable, and sellable flags will be defaulted to Y
  - Item number type will be defaulted to Manual - APCS will generate an item number using a pre-defined range of numbers 9000000000-9999999999.

> **Note:** As part of the implementation, a modification in the database should be made to the item number sequence generator to ensure that Oracle Retail Item Number type items numbers won't use this range. For cloud service implementations, this will require an SR be logged with cloud engineering.

  - Standard UOM will be defaulted to EA
  - Status will be defaulted to Worksheet
  - Store Order Multiple will default to eaches
  - Transaction level will default to 2.
- **Item Supplier** - At the item/supplier level, it will send just the supplier ID, but the following will also be defaulted in the template for this intersection:
  - Primary supplier flag = Y
  - Case name = CS (Case), Inner name = INR (Inner), Pallet name = PAL (Pallet)
- **Item UDA** - For this intersection, any list of value-type UDAs that have been associated with the placeholder item will be sent with their value.

Because the item data is still considered incomplete from a Merchandising perspective when it is received from APCS, it will be loaded into a staging area in Merchandising, where further details on the item can be defined for the style and to add sizes to allow SKUs to be created and approved. Once this occurs, and the full item details will be included in the Merchandise Hierarchy updates sent back to APCS and can be matched to the original placeholder styles.

> **Note:** For more information on Item Induction, see the white paper in the Merchandising Functional Library at My Oracle Support #1585843.1.

Description	Program
Item Induction	iindbatch.ksh (see also the Item Induction chapter above)

Integration to Oracle Retail Demand Forecasting Cloud Service (RDFCS)

Description	Program
Calendar Extract to Planning and Forecasting	BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB
Currency Rates Extract to Planning and Forecasting	BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB
Merchandise Hierarchy and Item Extract to Planning and Forecasting	BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB
Organization Hierarchy Extract to Planning and Forecasting	BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB
Store Extract to Planning and Forecasting	BDI_RPAS_Store_Fnd_PF_From_RMS_JOB

Description	Program
UDA Item Extract to Planning and Forecasting	BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB
Out of Stock Extract to Forecasting	BDI_RDF_StockOut_Tx_PF_From_RMS_JOB
Weekly Sales Extract to Forecasting	BDI_RDF_WeeklySales_Tx_PF_From_RMS_JOB

Integration from Oracle Retail Demand Forecasting Cloud Service (RDFCS)

Description	Program
Weekly/Daily Item Forecast Upload	load_item_forecast.ksh

Data Maintenance

Description	Program
Retail Item Forecast History	rms_oi_forecast_history.ksh
Forecast roll up refresh views	refreshmview.ksh
Purge Forecast Data	fcstprg
Purge Forecast Data	forecast_data_purge_job

Scheduling and dependency information for each program can be found in the program details below. Additional information about the flow of information between Merchandising and the Oracle Retail planning and forecasting applications can be found in the Retail Reference Architecture (available on My Oracle Support).The following processes support this integration.

Integration Program Summary

For additional details on integrating Merchandising with MFP and RDF, the *Oracle Retail Merchandise Financial Planning Operations Guide* provides information from the point of view of MFP and RDF.

The following processes support this integration.

Merchandise Hierarchy and Item Extract to Planning and Forecasting (BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB)

Module Name BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB

bdi_merch_extract_to_file_wrapper.sh

bdi_rpas_merchhier_extract.ksh

Description Extracts merchandise hierarchy and item information to RPAS

Functional Area Foundation

Module Type Integration

Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	ItemHdrAndMerchHier_Fnd_ProcessFlow_From_RMS ItemHdr_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This program extracts the merchandise hierarchy from company to transaction level item to planning and forecasting on a weekly basis. Additional key attributes about the items are also included, such as the primary supplier, brand, and any differentiators (for example, colors, sizes, and so on) that exist for the item.

Key assumptions for this integration:

- The full merchandise hierarchy and all items are sent each time this process is run.
- Only approved, inventoried and sellable transaction-level items will be included in the integration. Pack items are not included.
- All descriptions are sent in the primary language as defined in Merchandising.
- For transaction items that do not have a parent item, then the transaction item is also displayed as the parent item, as well as the parent/diff level.
- For a parent item that is not marked as an aggregate item or does not have any of its diffs flagged as aggregates, the parent item is sent as the parent/diff level for all of its transaction items.
- A single unit of measure is assumed for all items and therefore the standard units of measure for the items are not sent.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This program utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to the target applications. The batch job BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts Merch Hierarchy information
and writes it out to a flat file for processing by both MFP and RDF."/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
```

```
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RPAS_MerchHier_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (ItemHdrAndMerchHier_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to the target applications:

- Extractor jobs (MerchHier_Fnd_Extractor, ItemHdr_Fnd_Extractor) call respective BDI_MERCH_SQL and BDI_ITEM_SQL functions to extract data from Merchandising tables to BDI outbound staging tables MERCH_HIER_OUT and ITEM_HDR_OUT.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_merchhier_extract.ksh to write merchandise hierarchy and item information from the MERCH_HIER_OUT and ITEM_HDR_OUT tables into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - MFP_outboundLocation
  - RDF_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
COMPHEAD	Yes	No	No	No
DIVISION	Yes	No	No	No
GROUPS	Yes	No	No	No
DEPS	Yes	No	No	No
CLASS	Yes	No	No	No
SUBCLASS	Yes	No	No	No
ITEM_MASTER	Yes	No	No	No
DIFF_GROUP_HEAD	Yes	No	No	No
DIFF_IDS	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
MERCH_HIER_OUT	Yes	Yes	No	Yes
ITEM_HDR_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No
ITEM_SUPPLIER_OUT	Yes	Yes	No	Yes

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	The transaction level item ID.
ITEM_DESC	Char(250)	Yes	The transaction level item description.
ITEM_PARENT_DIFF	Char(30)	Yes	Concatenated value consisting of item parent ID with the composite diff aggregate. If there is no item parent, this will contain the transaction level item.
ITEM_PARENT_DIFF_DESC	Char(250)	Yes	Description of the item parent diff. Concatenated value consisting of the item parent description and the diff IDs for all diffs associated to the parent marked as aggregates. If there is no item parent, it will contain the transaction level item description.
ITEM_PARENT	Char(25)	Yes	If there is no item parent, it will contain the transaction level item.

Field Name	Field Type	Required	Description
ITEM_PARENT_DESC	Char(250)	Yes	If there is no item parent, it will contain the transaction level item description.
SUBCLASS_ID	Number(10)	Yes	Unique subclass ID
SUBCLASS_NAME	Char(120)	Yes	Concatenated value consisting of the subclass number with name.
CLASS_ID	Number(10)	Yes	Unique class ID
CLASS_NAME	Char(120)	Yes	Concatenated value consisting of the class number with name.
DEPT	Number(4)	Yes	Department ID
DEPT_NAME	Char(120)	Yes	Concatenated value consisting of the department ID and name.
GROUP_NO	Number(4)	Yes	Group ID
GROUP_NAME	Char(120)	Yes	Group name
DIVISION	Number(4)	Yes	Division ID
DIV_NAME	Char(120)	Yes	Division name
COMPANY	Number(4)	Yes	Company ID
COMPANY_NAME	Char(120)	Yes	Company name
FORECAST_IND	Char(1)	Yes	Indicates whether or not the item should be forecasted. Valid values are Y or N.
CLASS	Number(10)	Yes	The class ID that is displayed in the Merchandising screens.
SUBCLASS	Number(10)	Yes	The subclass ID that is displayed in the Merchandising screens.
BRAND_NAME	Char(30)	Yes	If a brand is not assigned, this is defaulted to 'NA'.
BRAND_DESCRIPTION	Char(120)	Yes	The brand description for the transaction item. If a brand is not assigned, this is defaulted to 'Not Assigned'.
SUPPLIER	Number(10)	Yes	The ID of the primary supplier for the transaction item.
SUPPLIER_NAME	Char(240)	Yes	The name of the primary supplier for the transaction item.
DIFF_1	Char(10)	No	The ID of the first diff for the transaction level item. If a diff is not assigned, this is defaulted to 'NA'.

Field Name	Field Type	Required	Description
DIFF_1_DESC	Char(120)	No	The name of the first diff for the transaction item. If a diff is not assigned, this is defaulted to 'unassigned'.
DIFF_2	Char(10)	No	The ID of the second diff for the transaction item. If a diff is not assigned, this is defaulted to 'NA'.
DIFF_2_DESC	Char(120)	No	The name of the second diff for the transaction item. If a diff is not assigned, this is defaulted to 'unassigned'.
DIFF_3	Char(10)	No	The ID of the third diff for the transaction item. If a diff is not assigned, this is defaulted to 'NA'.
DIFF_3_DESC	Char(120)	No	The name of the third diff for the transaction item. If a diff is not assigned, this is defaulted to 'unassigned'.
DIFF_4	Char(10)	No	The ID of the fourth diff for the transaction item. If a diff is not assigned, this is defaulted to 'NA'.
DIFF_4_DESC	Char(120)	No	The name of the fourth diff for the transaction item. If a diff is not assigned, this is defaulted to 'unassigned'.

Organizational Hierarchy Extract to Planning and Forecasting (BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB)

Module Name	BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB bdi_merch_extract_to_file_wrapper.sh bdi_rpas_orghier_extract.ksh
Description	Extracts organizational hierarchy information to RPAS
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	StoreAndWhAndOrgHier_Fnd_ProcessFlow_From_RMS Store_Fnd_Extractor Wh_Fnd_Extractor OrgHier_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This program extracts the organization hierarchy data from company to location, which can be stores or warehouses to planning and forecasting on a weekly basis. Additional key attributes about the organizational hierarchy will also be sent to assist in building alternate hierarchies for planning, such as channel.

Key assumptions for this integration:

- MFPCS will use the third level of the Merchandising hierarchy (area) to represent channel.
- The full organizational hierarchy is sent each time this process is run.
- All names and descriptions are sent in the primary language only.
- The location in the file can represent either a store or a virtual warehouse location.
- Because warehouses live outside the organization hierarchy, for the levels of the organizational hierarchy above location (chain through district) when the location is a warehouse, the warehouse ID and description will be repeated.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This program utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to the target applications. The batch job BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts Org Hierarchy information and
writes it out to a flat file for processing by both MFP and RDF."/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RPAS_OrgHier_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (StoreAndWhAndOrgHier_Fnd_ProcessFlow_From_RMS) to

perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor jobs (Store_Fnd_Extractor, Wh_Fnd_Extractor, OrgHier_Fnd_Extractor) call respective BDI_ORG_SQL functions to extract data from Merchandising tables to BDI outbound staging tables ORG_HIER_OUT, STORE_OUT, and WH_OUT.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_orghier_extract.ksh to write organization hierarchy information from the ORG_HIER_OUT, STORE_OUT, and WH_OUT tables into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - MFP_outboundLocation
  - RDF_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
STORE	Yes	No	No	No
WH	Yes	No	No	No
AREA	Yes	No	No	No
CHAIN	Yes	No	No	No
DISTRICT	Yes	No	No	No
REGION	Yes	No	No	No
COMPHEAD	Yes	No	No	No

Table	Select	Insert	Update	Delete
CHANNELS	Yes	No	No	No
CODE_DETAIL	Yes	No	No	No
STORE_FORMAT	Yes	No	No	No
LANG	Yes	No	No	No
VAT_REGION	Yes	No	No	No
TSFZONE	Yes	No	No	No
ORG_HIER_OUT	Yes	Yes	No	Yes
STORE_OUT	Yes	Yes	No	Yes
WH_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
LOCATION	Number(10)	Yes	Store or virtual warehouse ID
LOC_NAME	Char(150)	Yes	Store or warehouse name
DISTRICT	Number(10)	Yes	District ID; for warehouses, repeat the warehouse ID with the prefix "WH"
DISTRICT_NAME	Char(120)	Yes	District name; for warehouses, repeat the warehouse name
REGION	Number(10)	Yes	Region ID; for warehouses, repeat the warehouse ID with the prefix "WH"
REGION_NAME	Char(120)	Yes	Region name; for warehouses, repeat the warehouse name
AREA	Number(10)	Yes	Area ID; for warehouses, repeat the warehouse ID with the prefix "WH"
AREA_NAME	Char(120)	Yes	Area name; for warehouses, repeat the warehouse name
CHAIN	Number(10)	Yes	Chain ID; for warehouses, repeat the warehouse ID with the prefix "WH"
CHAIN_NAME	Char(120)	Yes	Chain name; for warehouses, repeat the warehouse name
COMPANY	Number(4)	Yes	Company ID
COMPANY_NAME	Char(120)	Yes	Company name

Field Name	Field Type	Required	Description
COMPANY_ CURRENCY	Char(3)	Yes	The currency code for the base currency defined in system options
LOC_TYPE	Char(1)	Yes	'S' for store, 'W' for warehouse
LOC_TYPE_NAME	Char(120)	Yes	Store or Warehouse depending on location type
PHYSICAL_WH	Number(10)	Yes	Physical warehouse ID for warehouses, repeat store ID for store
PHYSICAL_WH_ NAME	Char(120)	Yes	Physical warehouse name for warehouse, repeat store name for stores
CHANNEL_ID	Number(4)	Yes	Channel ID for the store or virtual warehouse; if no channel is defined, then NA
CHANNEL_NAME	Char(120)	Yes	Channel name; if no channel is defined, then 'unassigned'
STORE_CLASS	Char(1)	Yes	For stores, the store class ID; for warehouses or if no store class is defined; then NA.
STORE_CLASS_ DESCRIPTION	Char(250)	Yes	For stores, the description of the store class, if defined; for warehouses or if not defined for a store, then 'unassigned'.
STORE_FORMAT	Number(4)	Yes	For stores, the store format ID; for warehouses or if no store class is defined; then NA.
STORE_FORMAT_ NAME	Char(60)	Yes	For stores, the description of the store format, if defined; for warehouses or if not defined for a store, then 'unassigned'.

Store Extract to Planning and Forecasting (BDI_RPAS_Store_Fnd_PF_From_RMS_JOB)

Module Name	BDI_RPAS_Store_Fnd_PF_From_RMS_JOB bdi_merch_extract_to_file_wrapper.sh bdi_rpas_store_extract.ksh
Description	Extracts store information to RPAS
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI, shell scripts
Catalog ID	N/A

Runtime Parameters	Store_Fnd_ProcessFlow_From_RMS Store_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This program extracts store data to planning and forecasting on a weekly basis. This data supplements the store information included in the organizational hierarchy feed.

Key assumptions for this integration:

- Both stockholding and non-stockholding stores are included.
- Both company and franchise types of stores are included.
- All stores are sent each time this process is run.
- Planning will derive the status of the store (e.g. open or closed) based on the dates sent in this integration. For example, if the open date is in the past and there is no close date defined or it is a future date, then the store is considered open.
- All descriptions are sent in the primary language as defined in Merchandising.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This program utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement from Merchandising to the target applications.

The batch job BDI_RPAS_Store_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_Store_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts store information and writes
it out to a flat file for processing by both MFP and RDF."/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RPAS_Store_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (Store_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (Store_Fnd_Extractor) calls BDI_ORG_SQL.STORE_UP function to extract data from Merchandising tables to BDI outbound staging table STORE_OUT.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_store_extract.ksh to write store information from the STORE_OUT table into a comma-delimited flat file, which will be consumed by the target application. A zero-byte trigger file is also generated to signal that the extract process was successful. Two separate copies of the data file and the trigger file are sent to the target application.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - MFP_outboundLocation
  - RDF_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
STORE	Yes	No	No	No
CHANNELS	Yes	No	No	No
CODE_DETAIL	Yes	No	No	No
STORE_FORMAT	Yes	No	No	No
LANG	Yes	No	No	No

Table	Select	Insert	Update	Delete
VAT_REGION	Yes	No	No	No
TSFZONE	Yes	No	No	No
STORE_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
STORE	Number(10)	Yes	Store ID
STORE_NAME	Char(150)	Yes	Store name
DISTRICT	Number(10)	Yes	District in which the store is a member.
STORE_CLOSE_DATE	DATE	Yes	Date on which the store closed. If NULL, set to NA.
STORE_OPEN_DATE	DATE	Yes	Date on which the store opened
REMODEL_DATE	DATE	Yes	Date on which the store was last remodeled. If NULL, set to NA.
STORE_CLASS	Char(1)	Yes	ID for the store class of which the store is a member.
STORE_CLASS_DESCRIPTION	Char(250)	Yes	Store class description
STORE_FORMAT	Number(4)	Yes	Store format. If NULL, set to NA.
STORE_FORMAT_NAME	Char(60)	Yes	Store format name. If NULL, set to 'unassigned'.
CURRENCY	Char(3)	Yes	Currency under which the store operates.
STORE_TYPE	Char(6)	Yes	Indicates whether the store is a franchise (F) or company store (C).
STOCKHOLDING_IND	Char(1)	Yes	Indicates whether the store can hold stock. Valid values are Y or N.

Brand Extract to Planning (BDI_RPAS_Brand_Fnd_PF_From_RMS_JOB)

Module Name BDI_RPAS_Brand_Fnd_PF_From_RMS_JOB

bdi_merch_extract_to_file_wrapper.sh

bdi_rpas_brand_extract.ksh

Description	Extracts Brand information to Planning
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	Brand_Fnd_ProcessFlow_From_RMS Brand_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This process extracts its brand data to Planning on a weekly basis.

Key assumptions for this integration:

- The full set of brands is included in this integration each time it runs.
- Retailers will not create a Diff with an ID of 'BRAND'.
- In order to meet the format required by Planning, the UDA description in this extract is hard coded to "Brand" and does not take into account the primary language configuration in Merchandising.
- The intended targets for this integration are
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to Planning. The batch job BDI_RPAS_Brand_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_Brand_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
  <properties>
    <property name="description" value="Extracts Brand information and writes it
out to a flat file for processing by AP and IP."/>
  </properties>
  <step id="batchlet-step">
    <batchlet ref="BDIInvokerBatchlet">
      <properties>
        <property name="bdiProcessFlowUrl" value="#SysOpt.bdiProcessFlowUrl"/>
          <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
          <property name="predicateDS" value="RmsDBDS"/>
          <property name="predicateFunction" value="RMS_BATCH_STATUS_SQL.GET_EOW_
RUN_SIGNAL"/>
      </properties>
    </batchlet>
    <end on="COMPLETED"/>
  </step>
</job>
```

When the batch job BDI_RPAS_Brand_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_

BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (Brand_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (Brand_Fnd_Extractor) calls BDI_FOUNDATION_SQL.BRAND_UP function to extract data from Merchandising table BRAND to BDI outbound staging table BRAND_OUT.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_brand_extract.ksh to write brand information from the BRAND_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Two separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
BRAND	Yes	No	No	No
BRAND_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
UDA_ID	Char(6)	Yes	Hardcoded to 'BRAND'
UDA_DESC	Char(120)	Yes	Hardcoded to 'Brand'
BRAND_NAME	Char(30)	Yes	The brand ID from the Merchandising Brand table.
BRAND_DESCRIPTION	Char(120)	Yes	The brand description in the primary language from the Merchandising Brand table.

Calendar Extract to Planning and Forecasting (BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB)

Note: This module replaces the ftmednld.pc module from previous releases.

Module Name	BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB
Description	Extracts calendar information to RPAS from RMS
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job
Catalog ID	N/A
Runtime Parameters	Calendar_Fnd_ProcessFlow_From_RMS Calendar_Fnd_Extractor

Design Overview

This program extracts calendar data to planning and forecasting on a weekly basis.

Key assumptions for this integration:

- The last two years, current year, and two years into the future are extracted each time this process is run.
- A data set is sent each time the extract runs.
- This extract supports a 4-5-4 calendar only.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This program utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement from Merchandising to the target applications.

The batch job BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts calendar information and
writes it out to a flat file for processing by both MFP and RDF."/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RPAS_Calendar_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (Calendar_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (Calendar_Fnd_Extractor) calls BDI_FOUNDATION_ SQL.CALENDAR_UP function to extract data from Merchandising view V_BDI_ DAY_LEVEL_CALENDAR to BDI outbound staging table CALENDAR_OUT.
- A generic BDI Downloader file creator job writes calendar information from the CALENDAR_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - MFP_outboundLocation
  - RDF_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.

Schedule Information	Description
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
V_BDI_DAY_LEVEL_CALENDAR	Yes	No	No	No
CALENDAR_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
DAY	Date	Yes	The date for which the data was derived, in YYYYMMDD format
WEEK	Date	Yes	The end of week date for the day, in YYYYMMDD format
MONTH	Number(2)	Yes	The month number of the day in the year; valid values 1-12
QUARTER	Number(1)	Yes	The quarter of the year for the day; valid values 1-4
HALF	Number(1)	Yes	The half of the year for the day; valid values are 1 or 2
YEAR	Number(4)	Yes	The year for the day (YYYY format).
WEEK_OF_YEAR	Number(2)	Yes	The week of the year for the day; valid values 1-53
DAY_OF_WEEK	Number(1)	Yes	The day number within the week; valid values 1-7.

Currency Rates Extract to Planning and Forecasting (BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB)

Module Name	BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB bdi_merch_extract_to_file_wrapper.sh bdi_rpas_curr_conv_rates_extract.ksh
Description	Extracts currency rates information to RPAS
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	CurrConvRates_Fnd_ProcessFlow_From_RMS CurrConvRates_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This program extracts its currency rates data to planning and forecasting on a weekly basis.

Key assumptions for this integration:

- Only currency rates for which stores and warehouse exist will be included in the extract.
- Either the consolidated or operational rate will be sent based on the setting of the Consolidation system option. If Y, then the consolidation rates will be sent. If N, then the operational rates are used.
- All applicable currency rates are sent each time this process is run.
- The rates sent in this integration are based on a materialized view. The process that refreshes this view (batch_rfmvcurrconv.ksh) must be scheduled to ensure that the latest currency information is sent each week.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This program utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement from Merchandising to the target applications.

The batch job BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts currency conversion rate
```

```
information and writes it out to a flat file for processing by both MFP and
RDF."/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RPAS_CurrConvRates_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (CurrConvRates_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (CurrConvRates_Fnd_Extractor) calls BDI_FOUNDATION_SQL.CURR_CONV_RATES_UP function to extract data from Merchandising view MV_CURRENCY_CONVERSION_RATES to BDI outbound staging table CURR_CONV_RATES_OUT.
  - Only the currencies for which stores and warehouses exist in Merchandising will be extracted.
  - Either consolidated or operational rates will be included based on Merchandising system options (consolidation_ind).
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_curr_conv_rates_extract.ksh to write currency rates information from the CURR_CONV_RATES_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - MFP_outboundLocation
  - RDF_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date
Scheduling Considerations	N/A
Pre-Processing	batch_rfmvcurrconv.ksh
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
MV_CURRENCY_ CONVERSION_RATES	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
CURR_CONV_RATES_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_ MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_ DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
EFFECTIVE_DATE	Date	Yes	Holds the effective date of the exchange rate for the currencies and the exchange type
FROM_CURRENCY_ CODE	Char(3)	Yes	Holds the convert from currency code.
TO_CURRENCY_ CODE	Char(3)	Yes	Holds the convert to currency code.
EXCHANGE_TYPE	Char(1)	Yes	Identifies the type of exchange rate. This will be either C (consolidation) or O (operational).
EXCHANGE_RATE	Number(20,10)	Yes	Contains the exchange rate between the from and to currencies for the specified exchange type on the next effective date. It is expressed in terms of the to-currency.

Differentiator Extract to Planning (BDI_RPAS_Diff_Fnd_PF_From_RMS_JOB)

Module Name	BDI_RPAS_Diff_Fnd_PF_From_RMS_JOB bdi_merch_extract_to_file_wrapper.sh bdi_rpas_diff_extract.ksh
Description	Extracts Diff Types and Diff ID information to Planning
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	Diff_Fnd_ProcessFlow_From_RMS Diff_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This process extracts its differentiator data to Planning on a weekly basis.

Key assumptions for this integration:

- The full set of differentiators and diff types are included in this integration each time it runs.
- The intended targets for this integration are
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to Planning. The batch job BDI_RPAS_Diff_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_Diff_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
  <properties>
    <property name="description" value="Extracts Diff Types and Diff ID
information and writes it out to a flat file for processing by AP and IP."/>
  </properties>
  <step id="batchlet-step">
    <batchlet ref="BDIInvokerBatchlet">
      <properties>
        <property name="bdiProcessFlowUrl" value="#SysOpt.bdiProcessFlowUrl"/>
        <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
        <property name="predicateDS" value="RmsDBDS"/>
        <property name="predicateFunction" value="RMS_BATCH_STATUS_SQL.GET_EOW_
RUN_SIGNAL"/>
      </properties>
    </batchlet>
    <end on="COMPLETED"/>
  </step>
</job>
```

When the batch job BDI_RPAS_Diff_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (Diff_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (Diff_Fnd_Extractor) calls BDI_CROSS_PILLAR_SQL.DIFF_UP function to extract data from DIFF_IDS and DIFF_TYPE to BDI outbound staging table DIFF_OUT.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_diff_extract.ksh to write differentiator information from the DIFF_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
DIFF_IDS	Yes	No	No	No
DIFF_TYPE	Yes	No	No	No
DIFF_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
DIFF_TYPE_ID	Char(6)	Yes	The ID of the diff type (for example, C for color).
DIFF_TYPE_DESC	Char(120)	Yes	The description of the diff type (for example, Color) in the primary language.
DIFF_ID	Char(10)	Yes	The ID of the diff (for example, S for Small).
DIFF_DESC	Char(120)	Yes	The description of the diff (for example, Small) in the primary language.

Supplier Extract to Planning (BDI_RPAS_Supplier_Fnd_PF_From_RMS_JOB)

Module Name	BDI_RPAS_Supplier_Fnd_PF_From_RMS_JOB bdi_merch_extract_to_file_wrapper.sh bdi_rpas_supplier_extract.ksh
Description	Extracts Supplier information to Planning
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	Supplier_Fnd_ProcessFlow_From_RMS Supplier_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This process extracts supplier data to Planning on a weekly basis.

Key assumptions for this integration:

- All active, orderable supplier sites will be included in this integration each time it runs.
- Retailers will not create a Diff with an ID of 'SUP'.
- In order to meet the format required by Planning, the UDA description in this extract is hard coded to "Supplier" and does not take into account the primary language configuration in Merchandising.
- The intended targets for this integration are

  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to Planning.

The batch job BDI_RPAS_Supplier_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_Supplier_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
  <properties>
    <property name="description" value="Extracts Supplier information and writes
it out to a flat file for processing by AP and IP."/>
  </properties>
  <step id="batchlet-step">
    <batchlet ref="BDIInvokerBatchlet">
      <properties>
        <property name="bdiProcessFlowUrl" value="#SysOpt.bdiProcessFlowUrl"/>
        <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
        <property name="predicateDS" value="RmsDBDS"/>
        <property name="predicateFunction" value="RMS_BATCH_STATUS_SQL.GET_EOW_
RUN_SIGNAL"/>
      </properties>
    </batchlet>
    <end on="COMPLETED"/>
  </step>
</job>
```

When the batch job BDI_RPAS_Supplier_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (Supplier_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (Supplier_Fnd_Extractor) calls BDI_FOUNDATION_SQL.SUPS_UP function to extract data from the Merchandising table SUPS to BDI outbound staging table SUPS_OUT. Only supplier sites will be extracted.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_supplier_extract.ksh to write supplier information from the SUPS_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Two separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
SUPS	Yes	No	No	No
SUPS_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
UDA_ID	Char(6)	Yes	Hardcoded 'SUP'
UDA_DESC	Char(120)	Yes	Hardcoded 'Supplier'
SUPPLIER	Char(30)	Yes	The supplier site ID.
SUP_NAME	Char(120)	Yes	The supplier site name in the primary language.

UDA Extract to Planning (BDI_RPAS_UdaAndUdaValues_Fnd_PF_From_RMS_JOB)

Module Name BDI_RPAS_UdaAndUdaValues_Fnd_PF_From_RMS_JOB
bdi_merch_extract_to_file_wrapper.sh
bdi_rpas_uda_extract.ksh

Description Extracts LOV Type UDA information to Planning

Functional Area Foundation

Module Type Integration

Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	UdaAndUdaValues_Fnd_ProcessFlow_From_RMS Uda_Fnd_Extractor UdaValues_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This process extracts its UDA data to Planning on a weekly basis.

Key assumptions for this integration:

- The full set of user defined attributes (UDAs) is included in this integration each time it runs.
- Only list of value type UDAs will be included in the integration.
- The intended targets for this integration are
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to Planning. The batch job BDI_RPAS_UdaAndUdaValues_Fnd_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RPAS_UdaAndUdaValues_Fnd_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
  <properties>
    <property name="description" value="Extracts LOV Type UDA information and
writes it out to a flat file for processing by AP and IP."/>
  </properties>
  <step id="batchlet-step">
    <batchlet ref="BDIInvokerBatchlet">
      <properties>
        <property name="bdiProcessFlowUrl" value="#SysOpt.bdiProcessFlowUrl"/>
        <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
        <property name="predicateDS" value="RmsDBDS"/>
        <property name="predicateFunction" value="RMS_BATCH_STATUS_SQL.GET_EOW_
RUN_SIGNAL"/>
      </properties>
    </batchlet>
    <end on="COMPLETED"/>
  </step>
</job>
```

When the batch job BDI_RPAS_UdaAndUdaValues_Fnd_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an and-of-week date, it invokes a BDI process flow (UdaAndUdaValues_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor jobs (Uda_Fnd_Extractor, UdaValues_Fnd_Extractor) call respective BDI_FOUNDATION_SQL functions to extract data from Merchandising tables UDA and UDA_VALUES to BDI outbound staging tables UDA_OUT and UDA_VALUES_OUT.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rpas_uda_extract.ksh to write UDA information from the UDA_OUT and UDA_VALUES_OUT tables into a comma-delimited flat file, which will be consumed by the target applications. Only LOV type UDAs will be extracted. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
UDA	Yes	No	No	No
UDA_VALUES	Yes	No	No	No
UDA_OUT	Yes	Yes	No	Yes
UDA_VALUES_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
UDA_ID	Number(5)	Yes	The ID of the UDA assigned to the item.
UDA_DESC	Char(120)	Yes	The description of the UDA (for example, Fabric Content).
UDA_VALUE	Number(5)	Yes	The ID of the UDA value for the UDA assigned to the item.
UDA_VALUE_DESC	Char(250)	Yes	The description of the UDA value (for example, Cotton).

Inventory Extract to Planning (BDI_MFP_Inventory_Tx_PF_From_RMS_JOB)

Module Name	BDI_MFP_Inventory_Tx_PF_From_RMS_JOB
Description	Extracts inventory information to Planning
Functional Area	Inventory
Module Type	Integration
Module Technology	BDI job
Catalog ID	N/A
Runtime Parameters	Inventory_Tx_ProcessFlow_From_RMS Inventory_Tx_Extractor

Design Overview

This process extracts owned inventory information for inventoried, non-pack approved transaction items to planning on a weekly basis, at the end of the week. The integration captures the current on-hand and in-transit for all the included item/locations at the point in time that the integration is run.

Key assumptions for this integration:

- Only inventoried, approved transaction items are included in the integration.
- Any inventory for pack items is aggregated with inventory for the component items.
- Only stockholding stores are included in the integration.
- Cost values are based on system configuration for cost:
  - For a cost department with the system configured for average cost, the cost basis is the item/location's weighted average cost, converted to primary currency.
  - For a cost department with the system configured for standard cost, the cost basis is the item/locations unit cost, converted to primary currency.
  - For a retail department, the cumulative mark-on percentage is used to calculate cost based on the retail price, converted to primary currency.
- Retail values sent are based on the current item/location retail price, converted to primary currency. The retail will include VAT if the system option to include VAT

in the stock ledger is set to include VAT so that the retail values in this integration are consistent with other data sent to planning.

- All unit values are sent in terms of the standard unit of measure for the item.
- Planning will interpret inventory as being clearance if the clearance flag sent in this integration shows the item/location to be on clearance at the end of the week.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement from Merchandising to the target applications. The batch job BDI_MFP_Inventory_Tx_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_MFP_Inventory_Tx_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts information regarding
inventory for use by the MFP application"/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_MFP_Inventory_Tx_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (Inventory_Tx_ProcessFLow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (Inventory_Tx_ExtractorJob) calls BDI_MFP_SQL. INVENTORY_UP function to extract data from Merchandising view V_BDI_MFP_INVENTORY to BDI outbound staging table INVENTORY_OUT.
- A generic BDI Downloader file creator job writes inventory quantities information from the INVENTORY_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Two separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - MFP_outboundLocation

- AP_outboundLocation
- IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
V_BDI_MFP_INVENTORY	Yes	No	No	No
INVENTORY_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
EOW	Date	Yes	Indicates the end of week date that the on order information pertains to.
ITEM	Varchar2(25	Yes	Transaction level item only.
LOCATION	Number(10)	Yes	Could be a store or virtual warehouse.
LOC_TYPE	Varchar2(1)	Yes	Indicates if the location is a store or warehouse - S = Store; W = Warehouse.
CLEAR_IND	Number(1)	Yes	Indicates if the item/location is currently on clearance.

Field Name	Field Type	Required	Description
REGULAR_ INVENTORY_UNITS	Number(12,4)	Yes	Current owned inventory for the item/location in units based on the standard unit of measure; calculated as stock on hand + pack component stock on hand + in transit + pack component in transit.
REGULAR_ INVENTORY_COST	Number(20,4)	Yes	The cost value of current owned inventory for the item/location; calculated based on unit inventory and the cost basis of the item's department, as described above.
REGULAR_ INVENTORY_ RETAIL	Number(20,4)	Yes	The retail value of current owned inventory for the item/location; calculated based on the unit inventory value shown above and the current item/location unit retail.
UNIT_COST	Number(20,4)	Yes	The current supplier purchase cost for the item/location.
AV_COST	Number(20,4)	Yes	The current weighted average cost for the item/location.
UNIT_RETAIL	Number(20,4)	Yes	The current unit retail for the item/location. If the item is on clearance, this would be the clearance price.

OnOrder Extract to Planning (BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB)

Note: This module replaces the onordext.pc and onorddnld.pc modules from previous releases.

Module Name BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB

Description Extracts inventory information to Planning

Functional Area Inventory Tracking

Module Type Integration

Module Technology BDI job

Catalog ID N/A

Runtime Parameters OnOrder_Tx_ProcessFlow_From_RMS OnOrder_Tx_ Extractor

Design Overview

This process extracts its quantities on order to planning and forecasting on a weekly basis, at the end of the week. The integration sends any open on order quantities aggregated by week, grouped by the open to buy end of week date. Any on order quantity that is still open and has an OTB EOW date in the past will be combined with the current week's on order.

Key assumptions for this integration:

- Only orderable, inventoried, approved transaction items are included in the integration.
- Any on order for pack items is sent based on the component items.
- Purchase orders flagged to not be included in "on order" are not included in the integration.
- Cost and retail values sent are based on the purchase order's cost and retail value, converted to primary currency.
- Retail values will include VAT if the system option to include VAT in the stock ledger is set to include VAT so that the retail values in this integration are consistent with other data sent to planning.
- All unit values are sent in terms of the standard unit of measure for the item.
- Planning will interpret the on order as being clearance if the clearance flag sent in this integration shows the item/location to be on clearance at the end of the week.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to the target applications.

The batch job BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts information regarding
quantities on order for use by the MFP application"/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_MFP_OnOrder_Tx_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end of week date. If the vdate is an end of week date, it invokes a BDI process flow (OnOrder_Tx_ProcessFlow_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (OnOrder_Tx_Extractor) calls BDI_MFP_SQL. ON_ORDER_UP function to extract data from Merchandising view V_BDI_MFP_ON_ORDER to BDI outbound staging table ON_ORDER_OUT.
- A generic BDI Downloader file creator job writes quantities on order information from the ON_ORDER_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated locations as configured via BDI system options:
  - MFP_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
V_BDI_MFP_ON_ORDER	Yes	No	No	No
ON_ORDER_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
EOW	Date	Yes	Indicates the end of week date that the on order information pertains to.
ITEM	Varchar2(25)	Yes	Transaction level item only.
LOCATION	Number(10)	Yes	Could be a store or virtual warehouse.
LOC_TYPE	Varchar2(1)	Yes	Indicates if the location is a store or warehouse - S = Store; W = Warehouse.
CLEAR_IND	Number(1)	Yes	Indicates if the item/location is currently on clearance.
ON_ORDER_UNITS	Number(12)	Yes	Indicates the total quantity of the item in the order in standard unit of measure.
ON_ORDER_COST	Number(20,4)	Yes	on order * PO cost in primary currency
ON_ORDER_RETAIL	Number(20,4)	Yes	on order * PO retail in primary currency

Transaction Data Extract to Planning (BDI_MFP_TranData_Tx_PF_From_RMS_JOB)

Module Name	BDI_MFP_TranData_Tx_PF_From_RMS_JOB
Description	Extracts Transaction data to Planning from RMS
Functional Area	Transactional Data
Module Type	Integration
Module Technology	BDI job
Catalog ID	N/A
Runtime Parameters	TranData_Tx_ProcessFlow_From_RMS TranData_Tx_ Extractor

Design Overview

This process extracts transactional data to planning on a weekly basis, aggregating all transactions that posted in the last week, which could include transactions for previous weeks that posted late.

Key assumptions in this integration:

- Only orderable, inventoried, approved transaction items are included in the integration.
- Pack items are not included in this integration; any transactions involving pack items will be sent in terms of the pack's component items.
- Cost and retail values sent in primary currency.

- Sales sent will always be net sales. If gross sales are needed in Planning, then net sales can be combined with returns.
- Retail values will include VAT if the system option to include VAT in the stock ledger is set to include VAT so that the retail values in this integration are consistent with other data sent to planning.
- All unit values are sent in terms of the standard unit of measure for the item.
- Late posted transactions included in this integration may be for any week in the open stock ledger month, as well as any week in the previous month that posted during the week but before the previous month closed, if the month close ran during the current week.
- The intended targets for this integration are
  - Oracle Retail Merchandise Financial Planning Cloud Service (MFPCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to the target applications. The batch job BDI_MFP_TranData_Tx_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_MFP_TranData_Tx_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts information regarding
transaction data for use by the MFP application"/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_MFP_TranData_Tx_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (Trandata_Tx_ProcessFLow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (TranData_Tx_Extractor) calls BDI_MFP_SQL. TRAN_DATA_UP function to extract data from the Merchandising view V_BDI_MFP_TRAN_DATA to BDI outbound staging table TRAN_DATA_OUT.
- A generic BDI Downloader file creator job writes transactional information from the TRAN_DATA_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to

signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.

- The downloaded data files and trigger files are written to designated MFP location as configured via BDI system options:
  - MFP_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
V_BDI_MFP_TRAN_DATA	Yes	No	No	No
TRAN_DATA_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
EOW	Date	Yes	Indicates the end of week date that the information pertains to.
ITEM	Varchar2(25)	Yes	Transaction level item only.
LOCATION	Number(10)	Yes	Could be a store or virtual warehouse.
LOC_TYPE	Varchar2(1)	Yes	Indicates if the location is a store or warehouse - S = Store; W = Warehouse.

Field Name	Field Type	Required	Description
CLEAR_IND	Number(1)	Yes	If Y, item/location is currently on clearance.
NET_SALES_REG_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 1 and sales type = R
NET_SALES_REG_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 1 and sales type = R
NET_SALES_REG_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 1 and sales type = R
NET_SALES_PROMO_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 1 and sales type = P
NET_SALES_PROMO_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 1 and sales type = P
NET_SALES_PROMO_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 1 and sales type = P
NET_SALES_CLEAR_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 1 and sales type = C
NET_SALES_CLEAR_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 1 and sales type = C
NET_SALES_CLEAR_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 1 and sales type = C
NET_SALES_REG_RETAIL_VAT_EXCL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 2 and sales type = R
NET_SALES_PROMO_RTL_VAT_EXCL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 2 and sales type = P
NET_SALES_CLR_RETAIL_VAT_EXCL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 2 and sales type = C
RETURNS_REG_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 4 and sales type = R
RETURNS_REG_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 4 and sales type = R
RETURNS_REG_RETAIL	Number(20,4	No	tran_data_history.total_retail: tran_code = 4 and sales type = R
RETURNS_PROMO_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 4 and sales type = P

Field Name	Field Type	Required	Description
RETURNS_PROMO_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 4 and sales type = P
RETURNS_PROMO_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 4 and sales type = P
RETURNS_CLEAR_UNITS	Number(20,4)	No	tran_data_history.units: tran_code = 4 and sales type = C
RETURNS_CLEAR_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 4 and sales type = C
RETURNS_CLEAR_RETAIL	Number(20,4)	No	tran_data_history.total_cost: tran_code = 4 and sales type = C
REG_MARKDOWN_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code 13 - tran_code 14 (Markdown Cancel) - tran_code 11 (Markup)
PROMO_MARKDOWN_RETAIL_REG	Number(20,4)	No	tran_data_history.total_retail: tran_code = 15 - if the item is not on clearance EOW
PROMO_MARKDOWN_RETAIL_CLEAR	Number(20,4)	No	tran_data_history.total_retail: tran_code = 15 - if the item is on clearance EOW
CLEAR_MARKDOWN_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 16
WF_MARKDOWN_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 85
WF_MARKUP_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 84
SHRINK_UNITS	Number(12,4)	No	tran_data_history.units: tran_code 22
SHRINK_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code 22
SHRINK_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code 22
DEAL_INCOME_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code 6 & 7
RECEIPT_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 20 + 44
RECEIPT_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 20 + 44
RECEIPT_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 20 + 44

Field Name	Field Type	Required	Description
NON_SHRINK_ADJ_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 23
NON_SHRINK_ADJ_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 23
NON_SHRINK_ADJ_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 23
DEAL_INCOME_PURCHASES	Number(20,4)	No	tran_data_history.total_cost: tran_code 7
MARKUP	Number(20,4)	No	tran_data_history.total_retail: tran_code 11
MARKDOWN_CANCEL	Number(20,4)	No	tran_data_history.total_retail: tran_code 14
INTERCOMPANY_MARKUP	Number(20,4)	No	tran_data_history.total_retail: tran_code 17
INTERCOMPANY_MARKDOWN	Number(20,4)	No	tran_data_history.total_retail: tran_code 18
RTV_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 24
RTV_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 24
RTV_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 24
TSF_IN_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 30
TSF_IN_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 30
TSF_IN_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 30
TSF_IN_UNITS_BOOK	Number(12,4)	No	tran_data_history.units: tran_code = 31
TSF_IN_COST_BOOK	Number(20,4)	No	tran_data_history.total_cost: tran_code = 31
TSF_IN_RETAIL_BOOK	Number(20,4)	No	tran_data_history.total_retail: tran_code = 31
TSF_OUT_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 32
TSF_OUT_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 32
TSF_OUT_RETAIL	Number(20,4)	No	tran_data_history.total_retail: tran_code = 32
TSF_OUT_UNITS_BOOK	Number(12,4)	No	tran_data_history.units: tran_code = 33
TSF_OUT_COST_BOOK	Number(20,4)	No	tran_data_history.total_cost: tran_code = 33
TSF_OUT_RETAIL_BOOK	Number(20,4)	No	tran_data_history.total_retail: tran_code = 33

Field Name	Field Type	Required	Description
RECLASS_IN_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 34
RECLASS_IN_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 34
RECLASS_IN_ RETAIL	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 34
RECLASS_OUT_ UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 36
RECLASS_OUT_ COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 36
RECLASS_OUT_ RETAIL	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 36
TSF_IN_UNITS_ICT	Number(12,4)	No	tran_data_history.units: tran_code = 37
TSF_IN_COST_ICT	Number(20,4)	No	tran_data_history.total_cost: tran_code = 37
TSF_IN_RETAIL_ICT	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 37
TSF_OUT_UNITS_ ICT	Number(12,4)	No	tran_data_history.units: tran_code = 38
TSF_OUT_COST_ICT	Number(20,4)	No	tran_data_history.total_cost: tran_code = 38
TSF_OUT_RETAIL_ ICT	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 38
INTERCOMPANY_ MARGIN	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 39
TSF_RECEIPT_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 44
TSF_RECEIPT_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 44
TSF_RECEIPT_ RETAIL	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 44
RTV_RESTOCK_FEE	Number(20,4)	No	tran_data_history.total_cost: tran_code = 65
FRANCHISE_SALES_ UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 82
FRANCHISE_SALES_ COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 82
FRANCHISE_SALES_ RETAIL	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 82
FRANCHISE_ RETURNS_UNITS	Number(12,4)	No	tran_data_history.units: tran_code = 83
FRANCHISE_ RETURNS_COST	Number(20,4)	No	tran_data_history.total_cost: tran_code = 83
FRANCHISE_ RETURNS_RETAIL	Number(20,4)	No	tran_data_history.total_ retail: tran_code = 83
FRANCHISE_ RESTOCK_FEE	Number(20,4)	No	tran_data_history.total_cost: tran_code = 86

UDA Item Extract to Planning and Forecasting (BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB)

Module Name	BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB bdi_merch_extract_to_file_wrapper.sh bdi_rdf_itemuda_extract.ksh
Description	Extracts information for LOV type of UDAs to Planning and Forecasting
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job, shell scripts
Catalog ID	N/A
Runtime Parameters	ItemHdrAndUdaItemLov_Fnd_ProcessFlow_From_RMS ItemHdr_Fnd_Extractor UdaItemLov_Fnd_Extractor Database connection, download file location, filename, trigger filename

Design Overview

This process extracts user-defined attributes (UDAs) assigned to item to Planning and Forecasting on a weekly basis.

Key assumptions for this integration:

- Only list of value (LOV) type UDAs will be included.
- Both forecasted and non-forecasted items are included in this extract, with the forecast flag included.
- Planning and Forecasting can only support a specific UDA being associated with an item once. Merchandising has a configuration that allows the same UDA to be associated with an item more than one time. However, when implementing with Planning or Forecasting, this should be avoided for LOV-type UDAs to prevent issues with interpreting the data. If more than one is associated with the item, then only the last UDA with a particular ID will be visible in Planning and Forecasting.
- The intended targets for this integration are
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)
  - Assortment & Item Planning for Fashion/Softlines Cloud Service and Assortment & Item Planning Enterprise Edition Cloud Service (referred to jointly as APCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement from Merchandising to the target applications. The batch job BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts UDA item LOV information and
writes it out to a flat file for processing by RDF."/>
```

```
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RDF_UdaItemLov_Fnd_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (ItemHdrAndUdaItemLov_Fnd_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to the target applications:

- Extractor jobs (ItemHdr_Fnd_Extractor, UdaItemLov_Fnd_Extractor) call respective BDI_ITEM_SQL functions to extract data from Merchandising tables to BDI outbound staging tables ITEM_HDR_OUT and UDA_ITEM_LOV_OUT.
- Downloader file creator job calls the wrapper script, bdi_merch_extract_to_file_wrapper.sh, to set the runtime parameters on environment variables. This script will then call bdi_rdf_itemuda_extract.ksh to write LOV type of UDA information from the ITEM_HDR_OUT and UDA_ITEM_LOV_OUT tables into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. The data file and the trigger file are then sent to the target applications.
- The downloaded data file and trigger file are written to designated locations as configured through BDI system options:
  - RDF_outboundLocation
  - AP_outboundLocation
  - IP_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_MASTER	Yes	No	No	No
CLASS	Yes	No	No	No
SUBCLASS	Yes	No	No	No
DIFF_GROUP_HEAD	Yes	No	No	No
DIFF_IDS	Yes	No	No	No
SYSTEM_OPTION	Yes	No	No	No
UDA_ITEM_LOV	Yes	No	No	No
UDA	Yes	No	No	No
UDA_VALUES	Yes	No	No	No
UDA_ITEM_LOV_OUT	Yes	Yes	No	Yes
ITEM_HDR_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
ITEM	Char(25)	Yes	The ID of the item.
UDA_ID	Number(5)	Yes	The ID of the UDA assigned to the item.
UDA_DESC	Char(120)	Yes	The description of the UDA (for example, Fabric Content)
UDA_VALUE	Number(5)	Yes	The ID of the UDA value for the UDA assigned to the item.
UDA_VALUE_DESC	Char(250)	Yes	The description of the UDA value (for example, Cotton).
FORECAST_IND	Char(1)	Yes	Indicates whether or not the item is to be forecasted. Valid values are Y or N.

Out of Stock Extract to Forecasting (BDI_RDF_StockOut_Tx_PF_From_RMS_JOB)

Note: This module replaces the soutdnld.pc module from previous releases.

Module Name	BDI_RDF_StockOut_Tx_PF_From_RMS_JOB
Description	Extracts out of stock item location information to Forecasting
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job
Catalog ID	N/A
Runtime Parameters	StockOut_Tx_ProcessFlow_From_RMS StockOut_Tx_ Extractor

Design Overview

This process extracts items which are out of stock for use by Forecasting on a weekly basis. This integration sends all item/store combinations that meet the criteria for review and have a stock-on-hand position of less than or equal to zero at the end of the week.

Key assumptions for this integration:

- Only stockholding stores are included in this integration.
- Only forecasted items are included in this integration.
- Only item/store combinations that have a status of Active and a ranged flag of Yes are reviewed for stock out conditions.
- Only item/store combinations that have a last sold date that is between the end of week date and x number of days back are reviewed for stock out conditions, where x is the value reports system option value Days Since Last Transaction.
- The intended targets for this integration are
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to the target applications. The batch job BDI_RDF_StockOut_Tx_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RDF_StockOut_Tx_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts information for items which
are out of stock for use by the RDF application"/>
    </properties>
    <step id="batchlet-step">
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RDF_StockOut_Tx_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (StockOut_Tx_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (StockOut_Tx_ExtractorJob) calls BDI_RDF_SQL. STOCKOUT_UP function to extract data from the Merchandising view V_BDI_RDF_STOCKOUT to outbound staging table STOCKOUT_OUT.
- A generic BDI Downloader file creator job writes out of stock item information from the STOCKOUT_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful.
- The downloaded data files and trigger files are written to designated location as configured through BDI system options:
  - RDF_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
V_BDI_RDF_STOCKOUT	Yes	No	No	No
STOCKOUT_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
ITEM	Varchar2(25)	Yes	Item that is out of stock at the store.
STORE	Number(10)	Yes	Store that is out of stock for the item.
EOW_DATE	Date	Yes	Indicates the end of week date for which the data applies.
OUT_OF_STOCK	Number(1)	Yes	Flag to indicate if the item/store is out of stock at end of week. This will always be 1, as only out-of-stock items are sent.

Weekly Sales Extract to Forecasting (BDI_RDF_WeeklySales_Tx_PF_From_RMS_JOB)

Module Name	BDI_RDF_WeeklySales_Tx_PF_From_RMS_JOB
Description	Extracts weekly sales information to Forecasting
Functional Area	Foundation
Module Type	Integration
Module Technology	BDI job
Catalog ID	N/A
Runtime Parameters	WeeklySales_Tx_ProcessFlow_From_RMS WeeklySales_Tx_Extractor

Design Overview

This process extracts weekly sales for use by Forecasting on a weekly basis. It sends only the sales from the last week.

Key assumptions for this integration:

- This integration sends gross sales. Returns are not netted out of the sales values.
- Warehouse issues are not included in this integration. Only sales for stores.
- Only forecasted items are included in this integration.
- The intended targets for this integration are
  - Oracle Retail Demand Forecasting Cloud Service (RDFCS)

This process utilizes BDI (Bulk Data Integration) to facilitate the bulk data movement to the target applications.

The batch job BDI_RDF_WeeklySales_Tx_PF_From_RMS_JOB is defined in the Merchandising JOS batch job admin as follows:

```
<job id="BDI_RDF_WeeklySales_Tx_PF_From_RMS_JOB" version="1.0"
xmlns="http://xmlns.jcp.org/xml/ns/javaee">
    <properties>
        <property name="description" value="Extracts weekly sales information for
use by the RDF application"/>
    </properties>
    <step id="batchlet-step">
```

```
        <batchlet ref="BDIInvokerBatchlet">
            <properties>
                <property name="bdiProcessFlowUrl"
value="#SysOpt.bdiProcessFlowUrl"/>
                <property name="bdiProcessFlowCredential"
value="#SysOpt.bdiProcessFlowUrlUserAlias"/>
                <property name="predicateDS" value="RmsDBDS"/>
                <property name="predicateFunction" value="RMS_BATCH_STATUS_
SQL.GET_EOW_RUN_SIGNAL"/>
            </properties>
        </batchlet>
        <end on="COMPLETED"/>
    </step>
</job>
```

When the batch job BDI_RDF_WeeklySales_Tx_PF_From_RMS_JOB is executed, a batchlet (BDIInvokerBatchlet) starts the execution flow. It calls a PLSQL function (RMS_BATCH_STATUS_SQL.GET_EOW_RUN_SIGNAL) to ensure the process flow is only executed on an end-of-week date. If the vdate is an end-of-week date, it invokes a BDI process flow (WeeklySales_Tx_ProcessFlow_From_RMS) to perform a series of steps to extract, download, and transport the downloaded files to target applications:

- Extractor job (WeeklySales_Tx_ExtractorJob) calls BDI_RDF_SQL. WEEKLY_SALES_UP function to extract data from a Merchandising view V_BDI_RDF_WEEKLY_SALES to outbound staging table WEEKLY_SALES_OUT.
- A generic BDI Downloader file creator job writes weekly sales information from the WEEKLY_SALES_OUT table into a comma-delimited flat file, which will be consumed by the target applications. A zero-byte trigger file is also generated to signal that the extract process was successful. Separate copies of the data file and the trigger file are sent to the target applications.
- The downloaded data files and trigger files are written to designated location as configured via BDI system options:
  - RDF_outboundLocation

Scheduling Constraints

Schedule Information	Description
Processing Cycle	End of Day
Frequency	Scheduled daily but files will only be generated weekly on End of Week date.
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	N/A

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
V_BDI_RDF_WEEKLY_SALES	Yes	No	No	No
WEEKLY_SALES_OUT	Yes	Yes	No	Yes
BDI_DWNLDR_IFACE_MOD_DATA_CTL	Yes	No	No	No
BDI_DWNLDR_IFACE_DATA_CTL	Yes	No	No	No

Integration Contract

The flat file will contain the following information:

Field Name	Field Type	Required	Description
ITEM	Varchar2(25)	Yes	Indicates the item.
STORE	Number(10)	Yes	Indicates the store.
EOW_DATE	Date	Yes	Indicates the end of week date for which the data applies.
SALES_UNITS	Number(12,4)	No	This value will be the total sales units for the item/location for the week.
SALES_TYPE	Varchar2(1)	Yes	Indicates the sales type. For example, R (Regular Sales), P (Promotional Sales) or C (Clearance Sales).

Upload to RMS

The following module uploads data to Merchandising.

Weekly/Daily ItemForecast Upload (load_item_forecast)

Module Name load_item_forecast.ksh

Description Load daily/weekly item forecast from Oracle Retail Demand Forecasting (RDF) Cloud Service

Functional Area Integration - Forecast

Module Type Integration

Module Technology Ksh

Catalog ID N/A

Wrapper Script rmswrap_shell_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This script loads item forecast data into the Merchandising forecast tables.

The forecast data comes from Demand Forecasting in a CSV (comma separated) format file. Merchandising expects a single comma-delimited input file (that is, a csv file) in the format specified in the sqlldr control scripts load_item_forecast.ctl (for Weekly) and load_daily_item_forecast.ctl (for Daily). Please refer to the "Integration Contract" for more details. A run-time parameter (that is, run type) of 'D' or 'W' indicates whether the Daily or Weekly forecast data is being loaded into Merchandising. If the forecast is a daily forecast, information is written to the DAILY_ITEM_FORECAST table. If the forecast is a weekly forecast, information is written to the ITEM_FORECAST table. Depending on the run type parameter, the batch truncates the respective forecast table prior to loading.

Restart/Recovery

Evaluate the successful load of the data.

In case of any failures:

SQL load – SQL load dumps invalid records that do not meet certain technical requirements (that is, data type inconsistencies, and so on). The rejected record is written either to a bad file or to a discard file. The discard file contains records that do not satisfy conditions such as missing or invalid record types. Records with other technical issues are written to the bad file.

> **Note:** A non-fatal code is returned by the program and a message will be written to the log file if reject files are created.

User Action: When such conditions exist, you may update either the bad or the discard file and attempt to reload using the same files. You may also fix the data input file and reload, so that the item forecast tables will be truncated and upload item forecast tables with the corrected the data.

Integration Contract

If a run-time parameter of 'weekly' is used, the input file is a single comma-delimited file (that is, a CSV file):

Field Name	Field Type	Required	Description
EOW_DATE	Date(8)	Yes	Item_forecast.eow_date (YYYYMMDD)
ITEM	Char(25)	Yes	Item_forecast.item
LOC	Char(10)	Yes	Item_forecast.loc
FORECAST_SALES	Double(14)	Yes	Item_forecast.forecast_sales Note - this field can contain decimal quantities. Unlike quantity fields in Merchandising ProC Batch files, this qty field is not assumed to be extended to significant digits.

Field Name	Field Type	Required	Description
FORECAST_STD_DEV	Double(14)	Yes	Item_forecast.forecast_std_dev Note - this field can contain decimal quantities. Unlike quantity fields in Merchandising ProC Batch files, this qty field is not assumed to be extended to significant digits.

If a run-time parameter of 'daily' is used, the input file is a single comma-delimited file (that is, a CSV file):

Field Name	Field Type	Required	Description
DATA_DATE	Date(8)	Yes	Daily_item_forecast.data_date (YYYYMMDD)
ITEM	Char(25)	Yes	Daily_item _forecast.item
LOC	Char(10)	Yes	Daily_item _forecast.loc
FORECAST_SALES	Double(14)	Yes	Daily_item_forecast.forecast_sales Note - this field can contain decimal quantities. Unlike quantity fields in Merchandising ProC Batch files, this qty field is not assumed to be extended to significant digits.
FORECAST_STD_DEV	Double(14)	Yes	Daily_item_forecast.forecast_std_dev Note - this field can contain decimal quantities. Unlike quantity fields in Merchandising ProC Batch files, this qty field is not assumed to be extended to significant digits.

I/O Specification

N/A

Design Assumption

Domain is not a relevant concept any more. Domain_id on ITEM_FORECAST and DAILY_ITEM_FORECAST will always be 1.

Data Maintenance

The following modules maintain and clean up data.

Retain Item Forecast History (rms_oi_forecast_history.ksh)

Module Name	rms_oi_forecast_history.ksh
Description	Retain 4 weeks of Item Forecast History
Functional Area	Item Forecast, Inventory Analyst Report
Module Type	Admin
Module Technology	Ksh
Catalog ID	RMS491
Wrapper Script	rmswrap_shell.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program preserves four weeks of weekly forecasted sales data, moving the data from the item forecast table to the item forecast history table. The item forecast table is truncated and refreshed by the load_item_forecast.ksh weekly batch program.

The data in the history table is used to support the Inventory Variance to Forecast report in the Inventory Analyst dashboard. If the system is not configured to use this report, then running this batch job will NOT copy any data to the history table.

To support potentially large volume of data on the item forecast and item forecast history tables, the history table is interval partitioned by end of week date with a partition interval of 7 days and an interval high value of end of week date + 1. End of week date must be a valid end of week date based on calendar type - (4) 454 or (C) Standard Calendar.

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_FORECAST	Yes	No	No	No
ITEM_FORECAST_HIST	No	Yes	No	Yes

Design Assumptions

N/A

Purge Forecast Data (fcstprg)

Module Name	fcstprg.pc
Description	Purge Forecast Data
Functional Area	Interface - Planning
Module Type	Admin

Module Technology	ProC
Catalog ID	RMS227
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program deletes data from the Merchandising forecast information tables. This program serves to delete data by domains so that they can re-loaded with new forecast information from a forecasting system such as Demand Forecasting.

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
ITEM_FORECAST	No	No	No	Yes
DEPT_SALES_FORECAST	No	No	No	Yes
CLASS_SALES_FORECAST	No	No	No	Yes
SUBCLASS_SALES_ FORECAST	No	No	No	Yes

Design Assumptions

N/A

Purge Forecast Data (forecast_data_purge_job)

Module Name	forecast_data_purge_job
Description	Purge Forecast Data
Functional Area	Interface - Planning
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (FORECAST_DATA_PURGE_THREAD) will filter eligible records from RMS forecast information tables based on passed Domain ID. By default, all domains are captured and considered for purging criteria. These records

are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_FORECAST_DATA_PURGE_STG.

The Business logic program (FORECAST_DATA_PURGE) will process all records from the staging table. Using bulk processing, this program will delete the records from their respective RMS forecast information tables. Data deletion is performed by partition truncation, table truncation or deletion by domain. The method of deletion is dependent on whether or not the table is partitioned. This program serves to delete data by domains so that they can re-loaded with new forecast information from a forecasting system such as RDF. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by rownum

Restart/Recovery

NA

Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	Yes	No	No	No
B8D_FORECAST_DATA_PURGE_STG	Yes	Yes	No	Yes
ITEM_FORECAST	No	No	No	Yes
DEPT_SALES_FORECAST	No	No	No	Yes
CLASS_SALES_FORECAST	No	No	No	Yes
SUBCLASS_SALES_FORECAST	No	No	No	Yes

Design Assumptions

NA

27

Oracle Retail Sales Audit Batch Process and Designs

Sales Audit is subsystem of Merchandising. The purpose of Sales Audit is to accept transaction data from point-of-sale (POS) and Order-Management-System (OMS) applications and move the data through a series of processes that culminate in "clean" data. Data that Sales Audit finds to be inaccurate is brought to the attention of the retailer's sales auditors who can use the features of the sales audit system to correct the exceptions.

By using Sales Audit, you can quickly and accurately validate and audit transaction data before it is exported to other applications. Sales Audit uses several batch-processing modules to:

- Import POS/OMS transaction data sent from the store to the Sales Audit database
- Produce totals from user-defined totaling calculation rules that a user can review during the interactive audit
- Validate transaction and total data with user-defined audit rules that generate errors whenever data does not meet the criteria. You can review these errors during the interactive audit
- Create and export files in formats suitable for transfer to other applications
- Update the Sales Audit database with adjustments received from external systems on previously exported data

This chapter describes the batch processing modules involved with them.

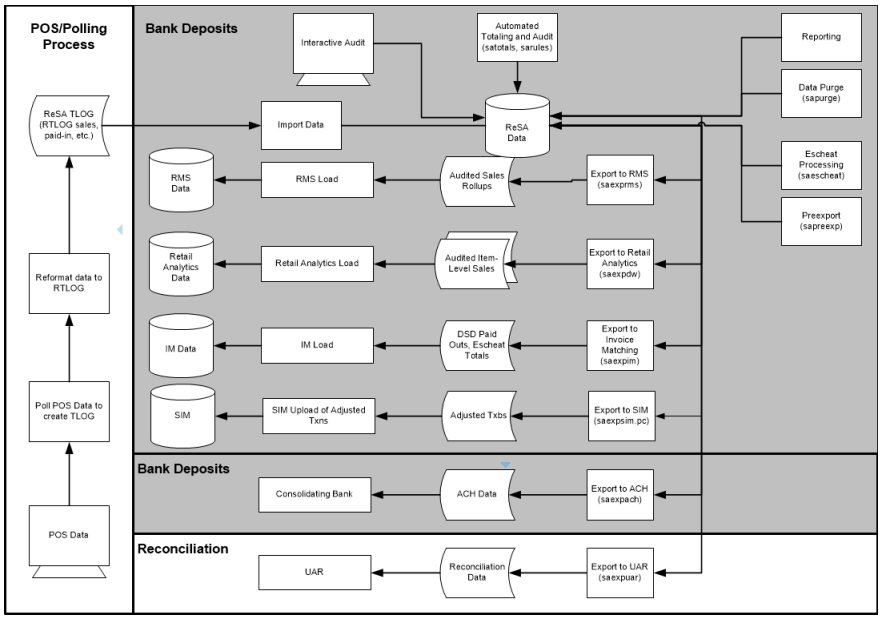
The term **store day** is used throughout this chapter. Store day describes all transactions that occur in one business day at one store or location. Because retailers need the ability to audit transactions on a store-by-store basis for a defined period of time, store day data is maintained separately, beginning with the initial import of data from the POS/OMS system.

Oracle Retail Sales Audit Dataflow Diagram

The following diagram illustrates how data flows within Sales Audit and between Sales Audit and other applications.

Note: All integrations are not depicted in this diagram.

Figure 27–1 Oracle Retail Sales Audit Dataflow Diagram



Oracle Retail Sales Import Process

Importing data from the POS to Sales Audit is a multi-step process that involves several Sales Audit batch processes.

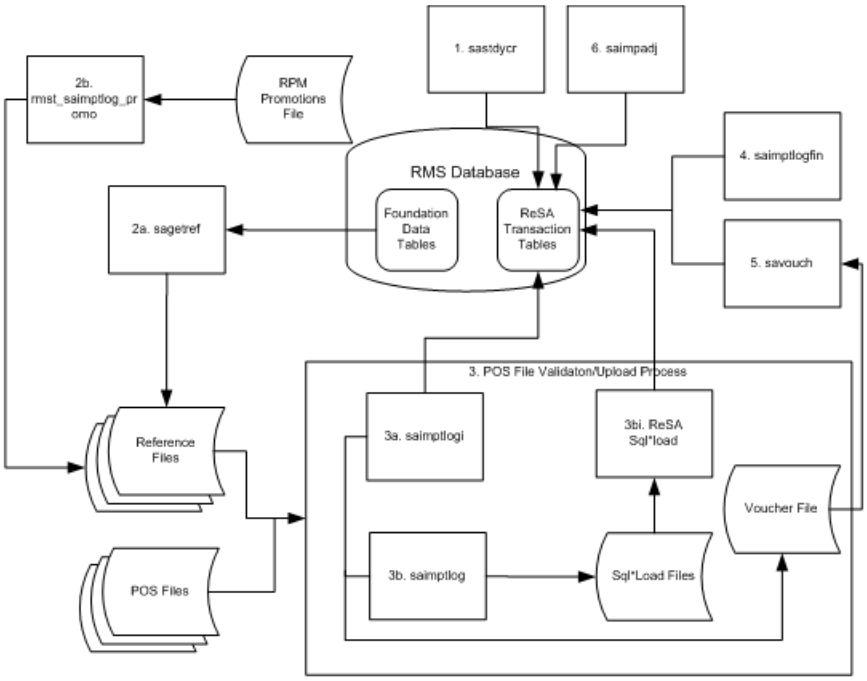
1. sastdycr.pc prepares the Sales Audit tables for data upload.
2. Reference file creation involves two processes:
   a. sagetref.pc creates a number of reference files to be used for validation in the File Validation/Upload Process.
      – One of the reference data files created by sagetref contains code values from the Merchandising CODE_HEAD and CODE_DETAIL tables. If the codes that Sales Audit uses are customized during the implementation, a library must be recompiled. This is discussed in detail in the design assumptions section of the sagetref program level information below.
      – The way primary variants are set up in Merchandising affects the data collected by sagetref and used in the File Validation/Upload Process. This is discussed in detail in the Design Assumptions section of the sagetref program level information below.
   b. rmst_saimptlog_promo.ksh transforms a promotions file from Pricing to the Sales Audit reference file format
3. The POS File Validation/Upload Process can be executed one of two sub-processes:
   a. saimptlogi.c validates files and uploads their transactions into the Sales Audit tables. This includes (as necessary) creating errors for the auditors to address
   b. saimptlog.c validates POS files and creates Sql*Loader Files. This includes (as necessary) creating errors for the auditors to address.
      – A Sql*Load process moves the transactions and errors into the Sales Audit tables.

c. Both saimptlog and saimptlogi create a voucher file to be used in later processing.

4. saimptlogfin.pc executes a number of import cleanup processes.
5. savouch.pc processes voucher sales and redemptions.
6. saimpadj.pc imports adjustments.

Each of the processes related to the import process are discussed in more detail at the program level later in this chapter.

Figure 27–2 Oracle Retail Sales Import Process



POS File Validation/Upload Sub-Process saimiptlog vs saimptlogi

saimptlogi.c and saimptlog.c perform the same business functions. Saimptlogi.c inserts directly into the database. Saimptlog uses SQL*Load to insert data. A retailer trickle polling or exporting a relatively small TLOG would be a good candidate to use saimptlogi.c. The detail discussion of these programs below contains more detail about the processing of these jobs.

Total Calculations and Rules

By providing additional values against which auditors can compare receipts, totaling is integral to the auditing process. Totaling also provides quick access to other numeric figures about the day's transactions.

Totaling in Sales Audit is dynamic. Sales Audit automatically totals transactions based on calculation definitions that the retailer's users create using the online Totals Calculation Definition Wizard. In addition, the retailer is able to define totals that come from the POS, but that Sales Audit does not calculate. Whenever you create new calculation definitions or edit existing ones, they become part of the automated totaling process the next time that satotals.pc runs.

Evaluating rules is also integral to the auditing process. Rules make the comparisons among data from various sources. These comparisons find data errors that could be the result of either honest mistakes or fraud. Finding these mistakes during the auditing process prevents these errors from being passed on to other systems, (for example, a merchandising system, a data warehouse system, and so on).

Like totaling, rules in Sales Audit are dynamic. They are not predefined in the system-retailers have the ability to define them through the online Rules Calculation Definition Wizard.

Errors uncovered by these rules are available for review during the interactive audit. Like satotals.pc, after users modify existing rules or create new ones, they become part of the rules the next time that sarules.pc runs.

Totals when Transactions are Modified

If a retailer modifies transactions during the Sales Audit interactive audit process, the totaling and auditing processes run again to recalculate store day totals. The batch module sapreexp.pc tracks all changed totals for the store day since the last export by comparing the latest prioritized version of each total defined for export with the version that was previously sent to each system. The module writes the changes to revision tables that the export modules later recognize as ready for export.

Oracle Retail Sales Export Process

Sales Audit prepares data for export to applications after:

- Some or all of the transactions for the day are imported (depending upon the application receiving Sales Audit's export).
- Totals have run.
- Audit rules have run.
- Errors in transactions and totals relevant for the system receiving the associated data are eliminated or overridden.

Depending upon the application, exported data consists of either transaction data or totals, or both. The process of exporting transaction data varies according to the unit of work selected in the Sales Audit system options. There are two units of work, transaction and store day. If the unit of work selection is transaction, Sales Audit exports transactions to Merchandising as soon as they are free of errors. If the unit of work selection is store day, transactions are not exported until all errors for that store day are either overridden or corrected.

Full Disclosure and Post-export Changes

If you modify data during the interactive audit that was previously exported to Merchandising, Sales Audit export batch modules re-export the modified data in accordance with a process called full disclosure. Full disclosure means that any previously exported values (dollars, units, and so on) are fully backed out before the new value is sent.

For example: a transaction originally shows a sale of 12 items, and that transaction is exported. During the interactive audit, you determine that the correct amount is 15 items, (where three are more than the original amount) and makes the change. Sales Audit then flags the corrected amount for export to the application.

The detail discussion of these programs below contains more detail about the processing of these jobs.

Batch Design Summary of Sales Audit Modules

The following list summarizes the Sales Audit batch modules that are involved with processing POS/OMS transaction data, audit totals and rules, exports to other applications, and modifications and adjustments.

Import Process Programs

- sastdycr.pc (Create Store Day for Expected Transactions)
- sagetref.pc (Get Reference Data for Sales Audit Import Processing)
- rmst_saimptlog_promo.ksh (Transform Promotion Reference File from Pricing format to Sales Audit Import Processing File Format)
- saimptlog.c/saimptlogi.c (Import of Unaudited Transaction data from POS to Sales Audit)
- saimptloglogtdup_upd (Processing to Allow Re-Upload of Deleted Transactions)
- saimptlogfin.pc (Complete Transaction Import Processing)
- savouch.pc (Sales Audit Voucher Upload)
- saimpadj.pc (Import Total Value Adjustments From External Systems to Sales Audit)

Totals/Rules Programs

- satotals.pc (Calculate Totals based on Client Defined Rules
- sa_totals_calc_job (Calculate Totals based on Client Defined Rules)
- sarules.pc (Evaluate Transactions and Totals based on Client Defined Rules)
- sa_rules_eval_job (Evaluate Transactions and Totals based on Client Defined Rules)

Export Programs

- sapreexp.pc (Prevent Duplicate Export of Total Values from Sales Audit)
- saexprms.pc (Export of POS transactions from Sales Audit to Merchandising)
- saordinvexp.pc (Export Inventory Reservation/Release for In Store Customer Order and Layaway Transactions from Sales Audit)
- saexpdw.pc (Export from Sales Audit to Oracle Retail Analytics)
- saexpsim.pc (Export of Revised Sale/Return Transactions from Sales Audit to SIM)
- saexpim.pc (Export DSD and Escheatment from Sales Audit to Invoice Matching)
- saexpgl.pc (Post User Defined Totals from Sales Audit to General Ledger)
- ang_saplgen.ksh (Extract of POS Transactions by Store/Date from Sales Audit for Web Search)
- saescheat.pc (Download of Escheated Vouchers from Sales Audit for Payment)
- saescheat_nextesn.pc (Generate Next Sequence for Escheatment Processing)

- saexpach.pc (Download from Sales Audit to Account Clearing House (ACH) System)
- saexpuar.pc (Export to Universal Account Reconciliation System from Sales Audit)

Other Programs

- saprepost.pc (Pre/Post Helper Processes for Sales Audit Batch Programs)
- sapurge.pc (Purge Aged Store/Day Transaction, Total Value, and Error Data from Sales Audit)
- b8d_sa_purge (Purge Into History Tables)

sastdycr (Create Store Day for Expected Transactions)

Module Name	sastdycr.pc
Description	Create Store Day for Expected Transactions
Functional Area	Oracle Retail Sales Audit
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RSA15
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The sastdycr batch program will create store/day, import log, and export log records. This program should run prior to uploading the sales data from POS/OMS for a given store/day. Store/days will be created for any open store expecting sales

Restart/Recovery

The logical unit of work in this program is store. Records are committed to the database when the commit counter is reached. The commit counter is defined by the value of INCREMENT_BY on the ALL_SEQUENCE table for the sequence SA_STORE_DAY_SEQ_NO_SEQUENCE.

Design Assumptions

N/A

sagetref (Get Reference Data for Sales Audit Import Processing)

Module Name	sagetref.pc
Description	Get Reference Data for Sales Audit Import Processing
Functional Area	Oracle Retail Sales Audit

Module Type	Integration
Module Technology	ProC
Catalog ID	RSA00
Wrapper Script	batch_sagetref.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will fetch all reference information needed by SAIMPTLOG.PC for validation purposes and write this information out to various output files. The following files are produced:

- Items - contains a listing of all items in the system.
- Wastage - contains information about all items that have wastage associated with them.
- Reference Items - contains reference items, or below transaction-level items.
- Primary Variant - contains primary variant information.
- Variable Weight UPC - contains all variable weight Universal Product Code (UPC) definitions in the system.
- Store/Days - contains all of the valid store/day combinations in the system.
- Codes and Code Types - contains all code types and codes used in field level validation.
- Error Codes and Descriptions - contains all error codes, error descriptions, and systems affected by the error.
- Store POS Mappings
- Tender Types
- Merchants
- Partners
- Suppliers
- Sales Audit Employees
- Banners
- Currency Codes
- Promotions
- Warehouses
- Inventory Statuses

These files will be used by the automated audit to validate information without repeatedly hitting the database.

When running sagetref.pc, retailers can either create and specify the output files, or create only the output that they desire. For example, a retailer interested in only creating a more recent employeefile would simply place a hyphen (-) in place of all the

other parameters, but still specify an employeefile name. This technique can be applied to as many or as few of the parameters as retailers wish. Note, however, that the item-related files (itemfile, refitemfile, wastefile, and primvariantfile) contain significant interdependence. Thus, item files must all be created or not created together.

In the list of reference data files above, standard UOM is part of the itemfile. To obtain the value, Sales Audit converts the selling Unit of Measure (UOM) to the standard UOM during batch processing. This conversion enables Sales Audit to later export the standard UOM to the systems that require its use.

Restart/Recovery

N/A

I/O Specification

Integration Type	Download from Merchandising
File Name	Determined by runtime parameter
Integration Contract	IntCon000113 (itemfile) IntCon000114 (wastefile) IntCon000115 (refitemfile) IntCon000116 (primvariantfile) IntCon000117 (varupcfile) IntCon000118 (storedayfile) IntCon000119 (promfile) IntCon000120 (codesfile) IntCon000121 (errorfile) IntCon000122 (storeposfile) IntCon000123 (tendertypefile) IntCon000124 (merchcodesfile) IntCon000125 (partnerfile) IntCon000126 (supplierfile) IntCon000127 (employeefile) IntCon000128 (bannerfile) IntCon000129 (promfile) IntCon000130 (whfile) IntCon000131 (invstatusfile)

File Name: Item File

The ItemFile file name (Itemfile) is not fixed; it is determined by a runtime parameter.

Table 27–1 Itemfile - File Layout

Field Name	Field Type	Default Value	Description
Item	Char(25)	N/A	Item number
Dept	Number(4)	N/A	Department ID
Class	Number(4)	N/A	Class
Subclass	Number(4)	N/A	Subclass ID
Standard UOM	Char(4)	N/A	Standard Unit of Measure
Catchweight Ind	Char(1)	N/A	Catch weight indicator
Class vat Ind	Char(1)	N/A	Class Vat Ind

File Name: Waste Data File

The Waste Data File file name (wastefile) is not fixed; it is determined by a runtime parameter.

Table 27–2 wastefile - File Layout

Field Name	Field Type	Default Value	Description
Item	Char(25)	N/A	Item number
Waste type	Char(6)	N/A	Waste type
Waste pct	Number(12,4)	N/A	Waste pct

File Name: Reference Item Data

The Reference Item Data file name (ref_itemfile) is not fixed; it is determined by a runtime parameter.

Table 27–3 Ref_itemfile - File Layout

Field Name	Field Type	Default Value	Description
Ref Item	Char(25)	N/A	Reference Item number
Item	Char(25)	N/A	Item number

File Name: Primary Variant Data File

The Primary Variant Data File file name (prim_variantfile) is not fixed; it is determined by a runtime parameter.

Table 27–4 prim_variantfile - File Layout

Field Name	Field Type	Default Value	Description
Location	Number(10)	N/A	Location number
Item	Char(25)	N/A	Item number
Prim Variant	Char(25)	N/A	Primary variant

File Name: Variable Weight UPC Definition File

The Variable Weight UPC Definition File file name (varupcfile) is not fixed; it is determined by a runtime parameter.

Table 27–5 varupcfile - File Layout

Field Name	Field Type	Default Value	Description
Format Id	Char(1)	N/A	Format ID
Format desc	Char(20)	N/A	Format description
Prefix length	Number(1)	N/A	Pefix Length
Begin item digit	Number(2)	N/A	Item digit begin
Begin var digit	Number(2)	N/A	Var digit begin
Check digit	Number(2)	N/A	Check digit
Default prefix	Number(1)	N/A	Default prefix
Prefix	Number(1)	N/A	Prefix

File Name: Valid Store/Day Combination File

The Valid Store/Day Combination File file name (storedayfile) is not fixed; it is determined by a runtime parameter.

Table 27–6 storedayfile - File Layout

Field Name	Field Type	Default Value	Description
Store	Number(10)	N/A	Store number
Business date	Char(8)	N/A	Business date in YYYYMMDD format
Store day seq no	Number(20)	N/A	Store day sequence number
Day	Number(3)	N/A	Day
Tran no generated	Char(6)	N/A	Generated transaction number
POS data expected	Char(1)	N/A	If system_code is POS, then Y; otherwise N
Currency rtl dec	Number(1)	N/A	Currency rtl dec
Currency code	Char(3)	N/A	Currency code
Country id	Char(3)	N/A	Country ID
Vat Include Ind	Char(1)	N/A	Vat Include Indicator

File Name: Codes File

The Codes File file name (codesfile) is not fixed; it is determined by a runtime parameter.

Table 27–7 codefile - File Layout

Field Name	Field Type	Default Value	Description
Code type	Char(4)	N/A	Code type
Code	Char(6)	N/A	Code ID
Code seq	Number(4)	N/A	Code sequence

File Name: Error Information File

The Error Information File file name (errorfile) is not fixed; it is determined by a runtime parameter.

Table 27–8 errorfile- File Layout

Field Name	Field Type	Default Value	Description
Error code	Char(25)	N/A	Error code
System Code	Char(6)	N/A	System Code
Error desc	Char(255)	N/A	Error description
Rec solution	Char(255)	N/A	Error rectify solution

File Name: Store POS Mapping File

The Store POS Mapping File file name (storeposfile) is not fixed; it is determined by a runtime parameter.

Table 27–9 storeposfile- File Layout

Field Name	Field Type	Default Value	Description
Store	Number(10)	N/A	Store
POS Type	Char(6)	N/A	Point Of Sale type
Start Tran No.	Number(10)	N/A	Start transaction number
End Tran No.	Number(10)	N/A	End transaction number

File Name: Tender Type Mapping File

The Tender Type Mapping File file name (tendertypefile) is not fixed; it is determined by a runtime parameter.

Table 27–10 tendertypefile - File Layout

Field Name	Field Type	Default Value	Description
Group	Char(6)	N/A	Tender type Group
Id	Number(6)	N/A	Tender type ID
Desc	Char(120)	N/A	Tender type description

File Name: Merchant Code Mapping File

The Merchant Code Mapping File file name (merchcodesfile) is not fixed; it is determined by a runtime parameter.

Table 27–11 merchcodesfile - File Layout

Field Name	Field Type	Default Value	Description
Non Merch Code	Char (6)	N/A	Non-Merchant Code

File Name: Partner Mapping File

The Partner Mapping File file name (partnerfile) is not fixed; it is determined by a runtime parameter.

Table 27–12 partnerfile - File Layout

Field Name	Field Type	Default Value	Description
Partner Type	Char(6)	N/A	Partner Type
Partner Id	Char(10)	N/A	Partner ID

File Name: Supplier Mapping File

The Supplier Mapping File file name (supplierfile) is not fixed; it is determined by a runtime parameter.

Table 27–13 supplierfile - File Layout

Field Name	Field Type	Default Value	Description
Supplier	Number(10)	N/A	Supplier ID
Sup status	Char(1)	N/A	Supplier status
Supplier Parent	Number(10)	N/A	Supplier Parent ID

File Name: Employee Mapping File

The Employee Mapping File file name (employeefile) is not fixed; it is determined by a runtime parameter.

Table 27–14 employeefile - File Layout

Field Name	Field Type	Default Value	Description
Store	Number(10)	N/A	Store ID
POS Id	Char(10)	N/A	Point Of Sale ID
Emp Id	Char(10)	N/A	Employee ID

File Name: Banner Information File

The Banner Information File file name (bannerfile) is not fixed; it is determined by a runtime parameter

Table 27–15 bannerfile - File Layout

Field Name	Field Type	Default Value	Description
Store	Number(10)	N/A	Store ID

Table 27–15 (Cont.) bannerfile - File Layout

Field Name	Field Type	Default Value	Description
Banner data	Number(4)	N/A	Banner ID
Stockholding Ind	Char(1)	N/A	Stockholding Indicator
Customer Order Loc Ind	Char(1)		Customer Order Location Indicator

File Name: Currency Information File

The Currency Information File file name (currencyfile) is not fixed; it is determined by a runtime parameter.

Table 27–16 currencyfile - File Layout

Field Name	Field Type	Default Value	Description
Currency Code	Char(1)	N/A	Currency Code

File Name: Promotion Information File

The Promotion Information File file name (promfile) is not fixed; it is determined by a runtime parameter.

Table 27–17 promfile - File Layout

Field Name	Field Type	Default Value	Description
Promotion	Number(10)	N/A	Promotion ID
Component	Number(10)	N/A	This contains the Offer ID value from Pricing.

File Name: Warehouse Information File

The Warehouse Information File filename (whfile) is not fixed; it is determined by a runtime parameter.

Table 27–18 whfile - File Layout

Field Name	Field Type	Default Value	Description
Warehouse	Number(10)	N/A	Warehouse ID
Physical Warehouse	Number(10)	N/A	Physical Warehouse ID
Customer Order Loc Ind	Char(1)	N/A	Customer Order Location Indicator

File Name: Inventory Status Information File

The Inventory Status Information File file name (invstatusfile) is not fixed; it is determined by a runtime parameter.

Table 27–19 invstatusfile - File Layout

Field Name	Field Type	Default Value	Description
Inventory Status	Char(10)	N/A	Inventory Status

Design Assumptions

N/A

A Note about Primary Variant Relationships

Depending upon a retailer's system parameters, the retailer designates the primary variant during item setup (through the front-end) for several reasons. One of the reasons is that, in some cases, an item may be identified at the POS by the item parent, but the item parent may have several variants.

The primary variant is established through a form at the item location level. The retailer designates which variant item is the primary variant for the current transaction level item. For more information about the new item structure in Merchandising, see the Oracle Retail Merchandising System User Guide.

In the example shown in the diagram below, the retailer has established their transaction level as an Item Level 2.

Note: The level of the primary variant is Item Level 1, and Item Level 3 is the sub-transaction level (the refitem).

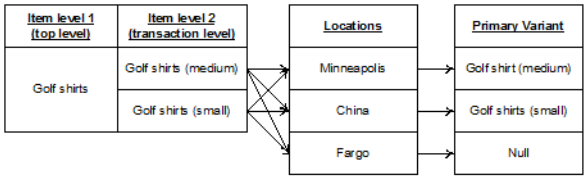
The retailer set up golf shirts in the merchandising system as its Item Level 1 above the transaction level. The retailer set up two items at level 2 (the transaction level) based on size (small and medium).

Note: The retailer assigned the level 2 items to all of the available locations (Minneapolis, China, and Fargo). The retailer also designated a primary variant for a single location - a medium golf shirt, in the case of Minneapolis, and a small golf shirt, in the case of China. The retailer failed to designate a primary variant for Fargo.

The primary variant affects Sales Audit in the following way. Sometimes a POS system does not provide Sales Audit with item level 2 (transaction item) data. For example, assume that the POS system in Minneapolis sold 10 medium golf shirts and 10 small golf shirts but only informed Sales Audit that 20 golf shirts were sold. 20 golf shirts presents a problem for Sales Audit because it can only interpret items at item level 2 (the transaction level). Thus, because medium golf shirts was the chosen primary variant for Minneapolis, the SAGETREF.PC module automatically transforms the 20 golf shirts into 20 medium golf shirts. If the same type of POS system in China informed Sales Audit of 20 golf shirts (instead of the 10 medium and 10 small that were sold), the sagetref.pc module would transform the 20 golf shirts sold in China into 20 small golf shirts. As the table shows, small golf shirts was the chosen primary variant for the China location. Sales Audit then goes on to export the data at the item 2 level (the transaction level) to, for example, a merchandising system, a data warehouse, and so on.

Note: Depending upon system parameters, if a retailer fails to set up the primary variant for a location, an invalid item error is generated during batch processing. In the example below, if the POS system in Fargo sold 10 medium golf shirts and 10 small golf shirts, but only informed Sales Audit that 20 golf shirts were sold, the sagetref.pc module would not have a way to transform those 20 golf shirts to the transaction level. Because Sales Audit can only interpret items above the transaction level in conjunction with a primary variant, the invalid item error would occur during batch processing.

Figure 27–3 Primary Variant Relationships



saimptlog/saimptlogi (Import of Unaudited Transaction Data from POS to Sales Audit)

Module Name	saimptlog.c saimptlogi.c
Description	Import of Unaudited Transaction data from POS to Sales Audit
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA11a RSA11b
Wrapper Script	batch_saimptlogi.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Importing POS and Order Management System (OMS) data to Sales Audit is a five or six-step process depending on whether saimptlogi or saimptlog is used. Saimptlog produces SQL*Loader files while saimptlogi does inserts directly into the database. Saimptlogi is meant for use in a trickle feed environment.

To import POS and OMS data, perform the following:

1. SAGETREF must be run to generate the current reference files:
   - Items

- Wastage
- Sub-transaction level items
- Primary variant relationships
- Variable weight PLU
- Store business day
- Code types
- Error codes
- Store POS
- Tender type
- Merchant code types
- Partner vendors
- Supplier vendors
- Employee ids
- Banner ids
- Currency File
- Promotions File
- Warehouse File
- Inventory Status File

These files are all used as input to SAIMPTLOG and SAIMPTLOGI. Because SAIMPTLOG and SAIMPTLOGI can be threaded, this boosts performance by limiting interaction with the database.

2. Either SAIMPTLOG or SAIMPTLOGI must be run against each file. The files are the transaction log files in Oracle Retail compatible format called RTLOG. The retailer is responsible for converting its transaction logs to RTLOGs. Both SAIMPTLOG and SAIMPTLOGI create a write lock, depending on the locking level specified in the Sales Audit System Options. It will create a write lock for a store/day combination on Sales Audit tables if the locking level indicated is Store Day. Otherwise, it will create a write lock for a transaction on Sales Audit tables if the locking level indicated is transaction. It will then set the data_status to loading until SAIMPTLOGFIN is executed. SAIMPTLOG generates distinct SQL*Loader files for that store/day for the sa_tran_head, sa_tran_item, sa_tran_disc, sa_tran_igtax (item Level Tax not VAT), sa_tran_payment (Payment details), sa_tran_tax, sa_tran_tender, sa_error, sa_customer, sa_cust_attrib, sa_tran_write_lock and sa_missing_tran tables, whereas SAIMPTLOGI inserts data to the database directly. Both produce an Oracle Retail formatted voucher file for processing.
3. SQL*Loader is executed to load the transaction tables from the files created by SAIMPTLOG. The store/day SQL*Loader files can be concatenated into a single file per table to optimize load times. Alternatively, multiple SQL*Loader files can be used as input to SQL*Loader. SQL*Loader may not be run in parallel with itself when loading a table. Header data (primary keys) must be loaded before ancillary data (foreign keys). This means that the sa_tran_head table must be loaded first, sa_tran_item before sa_tran_disc, and sa_customer before sa_cust_attrib. The remaining tables may be loaded in parallel.
4. SAVOUCH is executed to load each of the voucher files in Oracle Retail standard formatted. SAVOUCH may not be multi-threaded.

5. SAIMPTLOGFIN is executed to populate the sa_balance_group table, cancel post voided transactions and vouchers, validate missing transactions, and to mark the import as either partially or fully complete loaded. SAIMPTLOGFIN may not be multi-threaded.

Note: This design covers only Steps 2 and 3.

Restart and Recovery

N/A

File Upload Error Handling

For each RTLOG file, a record is written to the FILE_UPLOAD_STATUS table. In cases where a non-fatal error occurs after validating a record in the file, the error is written to the error file and a corresponding record is also inserted to the FILE_UPLOAD_ERRORS table.

I/O Specification

Integration Type Upload to Sales Audit

File Name Determined by runtime parameter

Integration Contract

Inputs from sagetref.pc:

IntCon000113 (itemfile)

IntCon000114 (wastefile)

IntCon000115 (refitemfile)

IntCon000116 (primvariantfile)

IntCon000117 (varupcfile)

IntCon000118 (storedayfile)

IntCon000119 (promfile)

IntCon000120 (codesfile)

IntCon000121 (errorfile)

IntCon000122 (storeposfile)

IntCon000123 (tendertypefile)

IntCon000124 (merchcodesfile)

IntCon000125 (partnerfile)

IntCon000126 (supplierfile)

IntCon000127 (employeefile)

IntCon000128 (bannerfile)

IntCon000129 (promfile)

IntCon000130 (whfile)

IntCon000131 (invstatusfile)

Inputs from POS:

IntCon000048 (RTLOG)

Outputs (if using saimptlog SQL Loader Option note that saimptlogi inserts directly into Sales Audit tables and does not create these output files)

IntCon000160 (SAVO)

IntCon000161 (satdisc.ctl)

IntCon000162 (saigtax.ctl)

IntCon000163 (sacust.ctl)

IntCon000164 (sathead.ctl)

IntCon000165 (satitem.ctl)

IntCon000166 (sattend.ctl)

IntCon000167 (satypmt.ctl)

IntCon000168 (samisstr.ctl)

IntCon000169 (sattax.ctl)

IntCon000170 (sacustatt.ctl)

IntCon000171 (saerror.ctl) (satwritelock.ctl)

The input files for this program are reference files generated by sagetref.pc and RTLOGs. Refer to the details for the sagetref.pc program for the input file specifications.

Output File Layout

Table 27–20 File Name: SAVO (Sales Audit Voucher File)

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File Type Record Descriptor	Char(5)	FHEAD	File type Record descriptor
	SA File Line No	Char(10)	N/A	Sales Audit File Line number
	Translator Id	Char(5)	SAVO	Identifies transaction type
	Sys Date	Char(14)	N/A	System date in YYYYMMDDHHMMSS format
	Is business date	Char(8)	N/A	Business date in YYYYMMDD format
FDETL	Record Descriptor	Char(5)	FDETL	File Type Record descriptor
	SA File Line No	Number(10)	N/A	Sales Audit File Line number
	Voucher seq Number	Number(20)	N/A	Unique identifier for an entry to sa_voucher table
	Voucher No	Char(25)	N/A	Voucher Number
	Voucher Type	Number(6)	N/A	Voucher Type
	Assigned Business Date	Char(8)	N/A	Business date in YYYYMMDD format
	Assigned Store	Number(10)	N/A	Store to which the voucher is assigned
	Issuing Date	Char(8)	N/A	Date this document was issued
	Issuing store	Number(10)	N/A	Store this document was issued from
	Issuing POS Register	Char(5)	N/A	Issuing Point Of Sale register
	Issuing Cashier	Char(10)	N/A	Issuing cashier
	Issued Tran Seq No.	Number(20)	N/A	Transaction sequence number
	Issued item seq number	Number(4)	N/A	Will hold the item sequence of the item when the voucher is sold as an item (gift voucher)

Table 27–20 (Cont.) File Name: SAVO (Sales Audit Voucher File)

Record Name	Field Name	Field Type	Default Value	Description
	Issued Tender Seq No.	Number(4)	N/A	Tender sequence number
	Issued Amount	Number(20)	N/A	Issued Amount * 10000 (4 implied digits)
	Issued Cust Name	Char(120)	N/A	Issued customer name
	Issued Customer Addr1	Char(240)	N/A	Issued customer addr1
	Issued Customer Addr2	Char(240)	N/A	Issued customer addr 2
	Issued Customer City	Char(120)	N/A	City of the customer, the voucher is issued
	Issued Customer State	Char(3)	N/A	State of the customer
	Issued Customer Postal Code	Char(30)	N/A	Postal address of the customer
	Issued Customer Country	Char(3)	N/A	Country of the customer the voucher was issued
	Recipient Name	Char(120)	N/A	Name of the intended recipient
	Recipient State	Char(3)	N/A	The state of the intended recipient
	Recipient Country	Char(3)	N/A	The country of the intended recipient
	Redemption Date	Char(8)	N/A	Date the voucher was redeemed
	Redemption Store	Number(10)	N/A	Store, the voucher was redeemed at
	Redemption Register	Char(5)	N/A	Register, the document was redeemed at
	Redemption cashier	Char(10)	N/A	Cashier redeeming the voucher
	Redemption tran seq number	Number(20)	N/A	Transaction number when the document was redeemed
	Redemption Tender seq number	Number(4)	N/A	This column will hold the tender sequence of the tender within the transaction when a voucher is redeemed as tender

Table 27–20 (Cont.) File Name: SAVO (Sales Audit Voucher File)

Record Name	Field Name	Field Type	Default Value	Description
	Redemption Amount	Number(20)	N/A	Amount the document was redeemed for*10000 (4 implied decimal places)
	Expiry Date	Char(8)	N/A	Expiry date
	Status	Char(1)	N/A	Indicator showing the document's status, issued or redeemed. Valid values = I - Issued, R - Redeemed
	Comments	Char(2000)	N/A	Comments
FTAIL	Record Descriptor	Char(5)	FTAIL	File Type Record descriptor
	SA File Line No.	Number(10)	N/A	Sales Audit File Line Number
	#lines	Number(10)	N/A	Total number of transaction lines in file (not including FHEAD and FTAIL)

Control Files

Table 27–21 File Name: Satdisc.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_DISC	TRAN_SEQ_NO	INTEGER EXTERNAL	20	1:20	N/A
	ITEM_SEQ_NO	INTEGER EXTERNAL	4	21:24	N/A
	DISCOUNT_SEQ_NO	INTEGER EXTERNAL	4	25:28	N/A
	RMS_PROMO_TYPE	CHAR	6	29:34	N/A
	PROMOTION	INTEGER EXTERNAL	10	35:44	N/A
	DISC_TYPE	CHAR	6	45:50	N/A
	COUPON_NO	CHAR	40	51:90	N/A
	COUPON_REF_NO	CHAR	16	91:106	N/A
	QTY	DECIMAL EXTERNAL	14	107:120	N/A
	UNIT_DISCOUNT_AMT	DECIMAL EXTERNAL	21	121: 141	N/A
	STANDARD_QTY	DECIMAL EXTERNAL	14	142:155	N/A

Table 27–21 (Cont.) File Name: Satdisc.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	STANDARD_UNIT_DISC_AMT	DECIMAL EXTERNAL	21	156:176	N/A
	REF_NO13	CHAR	30	177:206	N/A
	REF_NO14	CHAR	30	207:236	N/A
	REF_NO15	CHAR	30	237:266	N/A
	REF_NO16	CHAR	30	267:296	N/A
	ERROR_IND	CHAR	1	297:297	N/A
	CATCHWEIGHT_IND	CHAR	1	298:298	N/A
	UOM_QUANTITY	INTEGER EXTERNAL	12	299:310	N/A
	PROMO_COMP	INTEGER EXTERNAL	10	311:320	This field maps to the OFFER_ID field from Pricing
	STORE	INTEGER EXTERNAL	10	321:330	N/A
	DAY	INTEGER EXTERNAL	3	331:333	N/A

Table 27–22 File Name: Saigtax.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_IGTAX	TRAN_SEQ_NO	INTEGER EXTERNAL	20	1:20	N/A
	ITEM_SEQ_NO	INTEGER EXTERNAL	4	21:24	N/A
	IGTAX_SEQ_NO	INTEGER EXTERNAL	4	25:28	N/A
	TAX_AUTHORITY	CHAR	10	29:38	N/A
	IGTAX_CODE	CHAR	6	39:44	N/A
	IGTAX_RATE	DECIMAL EXTERNAL	11	45:65	N/A
	TOTAL_IGTAX_AMT	DECIMAL EXTERNAL	22	66:87	N/A
	STANDARD_QTY	DECIMAL EXTERNAL	14	88:101	N/A
	STANDARD_UNIT_IGTAX_AMT	DECIMAL EXTERNAL	21	102:122	N/A

Table 27–22 (Cont.) File Name: Saigtax.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	ERROR_IND	CHAR	1	123:123	N/A
	REF_NO_21	CHAR	30	124:153	N/A
	REF_NO_22	CHAR	30	154:183	N/A
	REF_NO_23	CHAR	30	184:213	N/A
	REF_NO_24	CHAR	30	214:243	N/A
	STORE	INTEGER EXTERNAL	10	244:253	N/A
	DAY	INTEGER EXTERNAL	3	254:256	N/A

Table 27–23 File Name: Sacust.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_ CUSTOMER	tran_seq_no	INTEGER EXTERNAL	20	1 :20	N/A
		DATE			
	cust_id	CHAR	16	21 :36	N/A
	cust_id_type	CHAR	6	37 :42	N/A
	name	CHAR	240	43 :162	N/A
	addr1	CHAR	240	163:402	N/A
	addr2	CHAR	240	403:642	N/A
	city	CHAR	240	643:762	N/A
	state	CHAR	3	763:765	N/A
	postal_code	CHAR	30	766:795	N/A
	country	CHAR	3	796:798	N/A
	home_phone	CHAR	20	799:818	N/A
	work_phone	CHAR	20	819:838	N/A
	e_mail	CHAR	100	839:938	N/A
	birthdate	DATE	8	939:946	Format is YYYYMMDD
	STORE	INTEGER EXTERNAL	10	947:956	N/A
	DAY	INTEGER EXTERNAL	3	957:959	N/A

Table 27–24 File Name: Sathead.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_ HEAD	Tran_seq_no	Integer external	20	1:20	N/A

Table 27–24 (Cont.) File Name: Sathead.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	Rev_no	Integer external	3	21:23	N/A
	Store_day_seq_no	Integer external	20	24:43	N/A
	Tran_datetime	date	14	44:57	Format is YYYYMMDDHH24MISS
	Register	char	5	58:62	N/A
	Tran_no	Integer external	10	63:72	N/A
	Cashier	char	10	73:82	N/A
	Salesperson	char	10	83:92	N/A
	Tran_type	char	6	93:98	N/A
	Sub_tran_type	char	6	99:104	N/A
	Orig_tran_no	Integer external	10	105:114	N/A
	Orig_reg_no	char	5	115:119	N/A
	Ref_no1	char	30	120:149	N/A
	Ref_no2	char	30	150:179	N/A
	Ref_no3	char	30	180:209	N/A
	Ref_no4	char	30	210:239	N/A
	Reason_code	char	6	240:245	N/A
	Vendor_no	char	10	246:255	N/A
	Vendor_invc_no	char	30	256:285	N/A
	Payment_ref_no	char	16	286:301	N/A
	Proof_of_delivery_no	char	30	302:331	N/A
	Status	char	6	332:337	N/A
	Value	char	22	338:359	Includes an optional negative sign and a decimal point
	Pos_tran_ind	char	1	360:360	N/A
	Update_id	char	30	361:390	N/A

Table 27–24 (Cont.) File Name: Sathead.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	Update_ datetime	date	14	391:404	Format is YYYYMM DDHH24MI SS
	Error_ind	char	1	405:405	N/A
	Banner_no	Integer external	4	406:409	N/A
	round_amt	Integer external	22	410:431	N/A
	ROUNDED_ OFF_AMT	INTEGER EXTERNAL	22	432:453	N/A
	CREDIT_ PROMOTIO N_ID	INTEGER EXTERNAL	10	454:463	N/A
	REF_NO25	CHAR	30	464:493	N/A
	REF_NO26	CHAR	30	494:523	N/A
	REF_NO27	CHAR	30	524:553	N/A
	STORE	INTEGER EXTERNAL	10	554:563	N/A
	DAY	INTEGER EXTERNAL	3	564:566	N/A
	RTLOG_ ORIG_SYS	CHAR	3	567:569	N/A
	TRAN_ PROCESS_ SYS	CHAR	3	570:572	N/A
	TRAN_ DATE	DATE	8	573:580	N/A
	REF_NO28	CHAR	30	581:610	N/A
	REF_NO29	CHAR	30	611:640	N/A
	REF_NO30	CHAR	30	641:670	N/A
	REF_NO31	CHAR	30	671:700	N/A

Table 27–25 File Name: Satitem.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_ ITEM	TRAN_SEQ_ NO	INTEGER EXTERNAL	20	1:20	N/A
	ITEM_SEQ_ NO	INTEGER EXTERNAL	4	21:24	N/A
	ITEM_ STATUS	CHAR	6	25:30	N/A
	ITEM_TYPE	CHAR	6	31:36	N/A
	ITEM	CHAR	25	37:61	N/A

Table 27–25 (Cont.) File Name: Satitem.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	REF_ITEM	CHAR	25	62:86	N/A
	NON_ MERCH_ ITEM	CHAR	25	87:111	N/A
	VOUCHER_ NO	CHAR	25	112:136	N/A
	DEPT	INTEGER EXTERNAL	4	137:140	N/A
	CLASS	INTEGER EXTERNAL	4	141:144	N/A
	SUBCLASS	INTEGER EXTERNAL	4	145:148	N/A
	QTY	DECIMAL EXTERNAL	14	149:162	Includes an optional negative sign and a decimal point
	UNIT_ RETAIL	DECIMAL EXTERNAL	21	163:183	Includes a decimal point
	UNIT_ RETAIL_ VAT_INCL	CHAR	1	184:184	Indicates whether unit retail includes or excludes VAT
	SELLING UOM	CHAR	4	185:188	N/A
	OVERRIDE_ REASON	CHAR	6	189:194	N/A
	ORIG_ UNIT_ RETAIL	DECIMAL EXTERNAL	21	195:215	Includes a decimal point
	STANDARD _ORIG_ UNIT_ RETAIL	DECIMAL EXTERNAL	21	216:236	N/A
	TAX_IND	CHAR	1	237:237	N/A
	ITEM_ SWIPED_ IND	CHAR	1	238:238	N/A
	ERROR_IND	CHAR	1	239:239	N/A
	DROP_ SHIP_IND	CHAR	1	240:240	N/A
	WASTE_ TYPE	CHAR	6	241:246	N/A

Table 27–25 (Cont.) File Name: Satitem.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	WASTE_PCT	DECIMAL EXTERNAL	12	247:258	Includes a decimal point
	PUMP	CHAR	8	259:266	N/A
	RETURN_ REASON_ CODE	CHAR	6	267:272	N/A
	SALESPERS ON	CHAR	10	273:282	N/A
	EXPIRATIO N_DATE	DATE	8	283:290	Format is YYYYMMD D
	STANDARD _QTY	DECIMAL EXTERNAL	14	291:304	Includes an optional negative sign and a decimal point
	STANDARD _UNIT_ RETAIL	DECIMAL EXTERNAL	21	305:325	Includes a decimal point
	STANDARD _UOM	CHAR	4	326:329	N/A
	REF_NO5	CHAR	30	330:359	N/A
	REF_NO6	CHAR	30	360:389	N/A
	REF_NO7	CHAR	30	390:419	N/A
	REF_NO8	CHAR	30	420:449	N/A
	CATCHWEI GHT_IND	CHAR	1	450:450	N/A
	SELLING_ ITEM	CHAR	25	451:475	N/A
	CUSTOMER _ORDER_ LINE_NO	INTEGER EXTERNAL	6	476:481	N/A
	MEDIA_ID	INTEGER EXTERNAL	10	482:491	N/A
	UOM_ QUANTITY	INTEGER EXTERNAL	12	492:503	N/A
	TOTAL_ IGTAX_AMT	DECIMAL EXTERNAL		504:524	N/A
	UNIQUE_ID	CHAR	25	525:652	N/A
	STORE	INTEGER EXTERNAL	10	653:662	N/A
	DAY	INTEGER EXTERNAL	3	663:665	N/A
	CUST_ ORDER_NO	CHAR	48	666:713	N/A

Table 27–25 (Cont.) File Name: Satitem.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	CUST_ ORDER_ DATE	DATE	14	714:727	Format is YYYYMMD DHH24MISS
	FULFILL_ ORDER_NO	CHAR	48	728:775	N/A
	NO_INV_ RET_IND	CHAR	1	776:776	N/A
	SALES_ TYPE	CHAR	1	777:777	N/A
	RETURN_ WH	INTEGER EXTERNAL	10	778:787	N/A
	RETURN_ DISPOSITIO N	CHAR	10	788:797	N/A
	ORIG_ STORE	INTEGER EXTERNAL	10	798:807	N/A
	ORIG_ TRAN_NO	INTEGER EXTERNAL	10	808:817	N/A
	FULFILLME NT_LOC_ TYPE	CHAR	2	818:820	N/A
	FULFILLME NT_LOC	INTEGER EXTERNAL	10	821:830	N/A

Table 27–26 File Name: Sattend.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_ TENDER	Tran_seq_no	Integer external	20	1:20	N/A
	Tender_seq_ no	Integer external	4	21:24	N/A
	Tender_ type_group	char	6	25:30	N/A
	Tender_ type_id	Integer external	6	31:36	N/A
	Tender_amt	decimal external	22	37:58	Includes an optional negative sign and a decimal point.
	Cc_no	Integer external	40	59:98	N/A
	Cc_exp_date	date	8	99:106	FORMAT IS YYYYMMD D
	Cc_auth_no	char	16	107:122	N/A

Table 27–26 (Cont.) File Name: Sattend.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	Cc_auth_src	char	6	123:128	N/A
	Cc_entry_ mode	char	6	129:134	N/A
	Cc_ cardholder_ verf	char	6	135:140	N/A
	Cc_term_id	char	5	141:145	N/A
	Cc_spec_ cond	char	6	146:151	N/A
	CC_TOKEN	CHAR	40	152:191	N/A
	Voucher_no	char	25	192:216	N/A
	Coupon_no	char	40	217:256	N/A
	Coupon_ref_ no	char	16	257:272	N/A
	CHECK_ ACCT_NO	CHAR	30	273:302	N/A
	CHECK_NO	INTEGER EXTERNAL	10	303:312	N/A
	IDENTI_ METHOD	CHAR	6	313:318	N/A
	IDENTI_ID	CHAR	40	319:358	N/A
	ORIG_ CURRENCY	CHAR	3	359:361	N/A
	ORIG_ CURR_AMT	DECIMAL EXTERNAL	22	362:383	N/A
	Ref_no9	char	30	384:413	N/A
	Ref_no10	char	30	414:443	N/A
	Ref_no11	char	30	444:473	N/A
	Ref_no12	char	30	474:503	N/A
	Error_ind	char	1	504:504	N/A
	STORE	INTEGER EXTERNAL	10	505:514	N/A
	DAY	INTEGER EXTERNAL	3	515:517	N/A

Table 27–27 File Name: Satpymt.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_ PAYMENT	TRAN_SEQ_ NO	INTEGER EXTERNAL	20	1:20	N/A
	PAYMENT_ SEQ_NO	INTEGER EXTERNAL	20	21:24	N/A

Table 27–27 (Cont.) File Name: Satpymt.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	PAYMENT_AMT	DECIMAL EXTERNAL	5	25:46	N/A
	ERROR_IND	CHAR	10	47:47	N/A
	STORE	INTEGER EXTERNAL	6	48:57	N/A
	DAY	INTEGER EXTERNAL	3	58:60	N/A

Table 27–28 File Name: Samisstr.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_MISSING_TRAN	MISS_TRAN_SEQ_NO	INTEGER EXTERNAL	20	1:20	N/A
	STORE_DAY_SEQ_NO	INTEGER EXTERNAL	20	21:40	N/A
	REGISTER	CHAR	5	41:45	N/A
	TRAN_NO	INTEGER EXTERNAL	10	46:55	N/A
	STATUS	CHAR	6	56:61	N/A
	RTLOG_ORIG_SYS	CHAR	3	62:64	N/A

Table 27–29 File Name: Sattax.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_TAX	TRAN_SEQ_NO	INTEGER EXTERNAL	20	1:20	N/A
	TAX_CODE	CHAR	6	21:26	N/A
	TAX_SEQ_NO	INTEGER EXTERNAL	4	27:30	N/A
	TAX_AMT	DECIMAL EXTERNAL	22	31:52	Includes an optional negative sign and a decimal point
	ERROR_IND	CHAR	1	53:53	N/A
	REF_NO17	CHAR	30	54:83	N/A
	REF_NO18	CHAR	30	84:113	N/A
	REF_NO19	CHAR	30	114:143	N/A
	REF_NO20	CHAR	30	144:173	N/A

Table 27–29 (Cont.) File Name: Sattax.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	STORE	INTEGER EXTERNAL	10	174:183	N/A
	DAY	INTEGER EXTERNAL	3	184:186	N/A

Table 27–30 File Name: Sacustatt.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_CUST_ ATTRIB	TRAN_SEQ_ NO	INTEGER EXTERNAL	20	1:20	N/A
	ATTRIB_ SEQSO	CHAR	4	21:24	N/A
	ATTRIB_ TYPE	CHAR	6	25:30	N/A
	ATTRIB_ VALUE	CHAR	6	31:36	N/A
	STORE	INTEGER EXTERNAL	10	37:46	N/A
	DAY	INTEGER EXTERNAL	3	47:49	N/A

Table 27–31 File Name: Saerror.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_ERROR	ERROR_ SEQ_NO	INTEGER EXTERNAL	20	1:20	N/A
	STORE_ DAY_SEQ_ NO	INTEGER EXTERNAL	20	21:40	N/A
	BAL_ GROUP_ SEQ_NO	INTEGER EXTERNAL	20	41:60	N/A
	TOTAL_ SEQ_NO	INTEGER EXTERNAL	20	61:80	N/A
	TRAN_SEQ_ NO	INTEGER EXTERNAL	20	81:100	N/A
	ERROR_ CODE	CHAR	25	101:125	N/A
	KEY_ VALUE_1	INTEGER EXTERNAL	4	126:129	N/A
	KEY_ VALUE_2	INTEGER EXTERNAL	4	130:133	N/A
	REC_TYPE	CHAR	6	134:139	N/A
	STORE_ OVERRIDE_ IND	CHAR	1	140:140	N/A

Table 27–31 (Cont.) File Name: Saerror.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
	HQ_ OVERRIDE_ IND	CHAR	1	141:141	N/A
	UPDATE_ID	CHAR	30	142:171	N/A
	UPDATE_ DATE TIME	DATE	14	172:185	Format is YYYYMMD DHH24MISS
	ORIG_ VALUE	CHAR	70	186:255	N/A
	STORE	INTEGER EXTERNAL	10	256:265	N/A
	DAY	INTEGER EXTERNAL	3	266:268	N/A

Table 27–32 File Name: Satwritelock.ctl

Table Name	Column Name	Field Type	Field Width	Position	Description
SA_TRAN_ WRITE_ LOCK	STORE_ DAY_SEQ_ NO	INTEGER EXTERNAL DATE	20	1:20	N/A
	TRAN_SEQ_ NO	INTEGER EXTERNAL DATE	20	21:40	N/A

Sales Audit Interface File Layout [rtlog]

The following illustrates the file layout format of the Oracle Retail TLOG. The content of each Oracle Retail TLOG file is per store per day. The filename convention is RTLOG_STORE_DATETIME.DAT (for example, RTLOG_1234_01221989010000.DAT).

Involves round off fields, credit promotion id, tax (vat) at item level and payment amount of customer orders.

Document has been modified regarding tender types, logic of handling both VAT-TAX in the system has been added.

Retailers must ensure that credit card numbers are masked when sent through RTLOGs. Similarly, when the tender type is check, checking account numbers must be masked when sent through RTLOGs. When Sales Audit encounters an RTLOG with a non-masked credit card or checking account number, the entire file will be rejected and will not be processed.

```
FHEAD      (Only 1 per file, required)
THEAD      (Multiple expected, one per transaction, required for each transaction)
TCUST      (Only 1 per THEAD record allowed, optional for some transaction types,
see table below)
CATT       (Attribute record specific to the TCUST record - Multiple allowed, only
valid if TCUST exists)
TITEM      (Multiple allowed per transaction, optional for some transaction types,
see table below)
IDISC      (Discount record specific to the TITEM record - Multiple allowed per
item, optional see table below)
```

```
IGTAX      (Vat/Tax record specific to the TITEM record - Multiple allowed per
item, optional. Either TTAX or IGTAX should appear in a given RTLOG, but not both,
see table below)
TTAX       (Vat/Tax record specific to the THEAD record - Multiple allowed per
transaction, optional. Either TTAX or IGTAX should appear in a given RTLOG, but
not both, see table below)
TPYMT      (Multiple allowed per transaction, will have the deposit amount for
pickup/delivery/layaway orders, optional see table below)
TTEND      (Multiple allowed per transaction, optional for some transaction types,
see table below)
TTAIL      (1 per THEAD, required)
FTAIL      (1 per file, required)
```

The order of the records within the transaction layout above is important. It aids processing by ensuring that the information is present when it is needed.

Fields expected in RTLog format based on the changes adopted -

	Version 16 Base RTLog	Version 16 with Customer Order Functionality	Version 16 with Additional reference fields in RTLog	Version 16 with Customer Order Functionality and with Additional reference fields in RTLog
Reference No. 28	N	N	Y	Y
Reference No. 29	N	N	Y	Y
Reference No. 30	N	N	Y	Y
Reference No. 31	N	N	Y	Y
Fulfillment Location type	N	Y	N	Y
Fulfillment Location	N	Y	N	Y

Table 27–33 File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	File Type Definition	Char(4)	RTLG	Identifies file as Oracle Retail TLOG.	Y	Left/Blank
	File Create Date	Char(14)	Create date	Date and time file was written by external system (YYYYMMDD DHHMMSS).	Y	Left/None

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Business Date	Char(8)	Business Date to process	Business date of transactions (YYYYMMDD).	Y	Left/None
	Location Number	Char(10)	Specified by external system	Store or warehouse identifier.	Y	Left/None
	Reference Number	Char(30)	Specified by external system	This may contain the Polling ID associated with the consolidated TLOG file or used for other purpose.	N	Left/Blank
	RTLOG Originating System	Char(3)	POS	Identifies the system the RTLOG file originated from. Valid values are OMS and POS.	Y	Left/None
Transaction Header	File Type Record Descriptor	Char(5)	Char(5) THEAD	Identifies file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	Register	Char(5)	Transaction date	Till used at the store.	Y	Left/Blank
	Transaction Date	Char(14)	N/A	Date for the transactions that were processed at the POS (YYYYMMDDHHMMSS).	Y	Left/None

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Transaction Number	Number(10)	N/A	Transaction identifier. If sa_system_options, wkstation_tran_append_ind is Y, then the first 3 digits indicate the workstation ID and last 7 digits indicate the transaction number.	Y	Right/0
	Cashier	Char(10)	N/A	Cashier identifier.	N	Left/Blank
	Salesperson	Char(10)	N/A	Salesperson identifier.	N	Left/Blank
	Transaction Type	Char(6)	Refer to TRAT code_type for a list of valid types.	Transaction type.	Y	Left/Blank
	Sub-transaction type	Char(6)	Refer to TRAS code_type for a list of valid types.	Sub-transaction type. For sale, it can be employee, drive-off, and so on.	N	Left/Blank
	Orig_tran_no	Number(10)	N/A	Populated only for post-void transactions. Transaction number for the original transaction that will be cancelled.	N	Right/0
	Orig_reg_no	Char(5)	N/A	Populated only for post-void even exchange and return transactions. Register number from the original transaction	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Reason Code	Char(6)	Refer to REAC code_type for a list of valid codes. If the transaction type is PAIDOU and the sub transaction type is MV or EV, than the valid codes come from the non_merch_code_head table.	Reason entered by the cashier for some transaction types. Required for Paid In and Paid out transaction types, but can also be used for voids, returns, and so on.	N	Left/Blank
	Vendor Number	Char(10)	N/A	Supplier ID for a merchandise vendor paid out transaction; partner ID for an expense vendor paid out transaction.	N	Left/Blank
	Vendor Invoice Number	Char(30)	N/A	Invoice number for a vendor paid out transaction.	N	Left/Blank
	Payment Reference Number	Char(16)	N/A	The reference number of the tender used for a vendor payout. This could be the money order number, check number, and so on.	N	Left/Blank
	Proof of Delivery Number	Char(30)	N/A	Proof of receipt number given by the vendor at the time of delivery. This field is populated for a vendor paid out transaction.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Reference Number 1	Char(30)	Na	Number associated with a particular transaction, for example, whether for a Store Conditions transaction. The SA_ REFERENCE table defines what this field can contain for each transaction type.	N	Left/Blank
	Reference Number 2	Char(30)	N/A	Char(30)	N	Left/Blank
	Reference Number 3	Char(30)	N/A	Third generic reference number.	N	Left/Blank
	Reference Number 4	Char(30)	N/A	Fourth generic reference number.	N	Left/Blank
	Value Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes.	Sign of the value.	Y if Value is present.	Left/None
	Value	Number(20)	N/A	Value, with 4 implied decimal places. Populated by the retailer for TOTAL transaction, populated by Sales Audit for SALE and RETURN transactions.	Y if tran is a TOTAL	Right/0 when value is present. Blank when no value is sent.
	Banner id	Number(4)	N/A	Banner ID of the location.	Y	Right/0 when value is present. Blank when no value is sent

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Rounded Amount Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes.	Sign of rounded amount. Amount Sign is not used.	Y	Left/None
	Rounded Amount	Number(20)	N/A	Total rounded amount, with 4 implied decimal places. Rounded Amount is not used.	Y	Right/0 when RoundedAmount is present otherwise blank
	Rounded Off Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes.	Rounded Off Sign is not used.	Y	Left/None
	Rounded Off Amount	Number(20)	N/A	Rounded off amount, with 4 implied decimal places. Rounded Off Amount is not used.	Y	Right/0 when RoundedAmount is present otherwise blank
	Credit Promotion Id	Char(10)	N/A	Credit Promotional ID.	Y	Left/None
	Reference Number 25	Char(30)	N/A	N/A	N	Left/Blank
	Reference Number 26	Char(30)	N/A	N/A	N	Left/Blank
	Reference Number 27	Char(30)	N/A	N/A	N	Left/Blank
	Transaction Processing System	Char(3)	Valid values are OMS and POS.	Contains the ID of the system that processed the transaction.	N	Left/None
	Reference Number 28	Char(30)		Generic Reference Number. It can be ignored if not needed based on the type of RTLOG being used.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Reference Number 29	Char(30)		Generic Reference Number. It can be ignored if not needed based on the type of RTLOG being used.	N	Left/Blank
	Reference Number 30	Char(30)		Generic Reference Number. It can be ignored if not needed based on the type of RTLOG being used.	N	Left/Blank
	Reference Number 31	Char(30)		Generic Reference Number. It can be ignored if not needed based on the type of RTLOG being used.	N	Left/Blank
Transaction Customer	File Type Record Descriptor	Char(5)	TCUST	Identifies the file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file	Y	Right/0
	Customer ID	Char(16)	Customer identifier	The ID number of a customer.	Y	Left/Blank
	Customer Type ID	Char(6)	Refer to CIDT code_ type for a list of valid types.	Customer ID type.	Y	Left/Blank
	Customer Name	Char(120)	N/A	Customer name.	N	Left/Blank
	Address 1	Char(240)	N/A	Customer address.	N	Left/Blank
	Address 2	Char(240)	N/A	Additional field for customer address.	N	Left/Blank
	City	Char(120)	N/A	City.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	State	Char(12)	State identifier	State.	N	Left/Blank
	Zip Code	Char(30)	Zip identifier	Zip code.	N	Left/Blank
	Country	Char(3)	N/A	Country.	N	Left/Blank
	Home Phone	Char(20)	N/A	Telephone number at home.	N	Left/Blank
	Work Phone	Char(20)	N/A	Telephone number at work.	N	Left/Blank
	E-mail	Char(100)	N/A	E-mail address.	N	Left/Blank
	Birthdate	Char(8)	N/A	Date of birth. (YYYYMMDD)	N	Left/Blank
Customer Attribute	File Type Record Descriptor	Char(5)	CATT	Identifies file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	Attribute type	Char(6)	Refer to SACA code_type for a list of valid types.	Type of customer attribute	Y	Left/Blank
	Attribute value	Char(6)	Refer to members of SACA code_type for a list of valid values.	Value of customer attribute.	Y	Left/Blank
Transaction Item	File Type Record Descriptor	Char(5)	TITEM	Identifies file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Item Status	Char(6)	Refer to SASI code_ type for a list of valid codes.	Status of the item within the transaction. Valid values are: V - Void item S - Sold item R - Returned item ORI - Order Initiate ORC - Order Cancel ORD - Order Complete LIN - Layaway Initiate LCA - Layaway Cancel LCO - Layaway Complete	Y	Left/Blank
	Item Type	Char(6)	Refer to SAIT code_ type for a list of valid codes.	Identifies what type of item is transmitted.	Y	Left/Blank
	Item number type	Char(6)	Refer to UPCT code_ type for a list of valid codes.	Identifies the type of item number if the item type is ITEM or REF	N	Left/Blank
	Format ID	Char(1)	VPLU format ID	Used to interpret VPLU items.	N	Left/Blank
	Item	Char(25)	Item identifier	Identifies the merchandise item.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Reference Item	Char(25)	Item identifier	Identifies the sub-transaction level merchandise item.	N	Left/Blank
	Non-Merchandise Item	Char(25)	Item identifier	Item identifier Identifies a non-merchandise item.	N	Left/Blank
	Voucher	Char(25)	N/A	Gift certificate number.	N	Right/0
	Department	Number(4)	N/A	Identifies the department to which this item belongs. This is filled in by saimptlog.	N	Right/Blank
	Class	Number(4)	Class of the item	Class of item sold or returned. Not required from a retailer; populated by Sales Audit. This is filled in by saimptlog.	N	Right/Blank
	Subclass	Number(4)	Subclass of the item	Subclass of the item sold or returned. Not required from a retailer; populated by Sales Audit. This is filled in by saimptlog.	N	Right/Blank
	Quantity Sign	Char(1)	Refer to SIGN code_type for a list of valid codes.	Sign of the quantity	Y	Left/None

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Quantity	Number(12)	N/A	Number of items purchased, with 4 decimal places.	Y	Right/0
	Selling Unit of Measure	Char(4)	N/A	Unit of measure of the item's quantity.	Y	Left/None
	Unit Retail	Number(20)	N/A	Unit retail, with 4 implied decimal places.	Y	Right/0
	Override Reason	Char(6)	Refer to ORRC code_type for a list of valid codes.	This column is populated when an item's price has been overridden at the POS to define why it was overridden.	Y if unit retail was manually entered	Left/Blank
	Original Unit Retail	Number(20)	N/A	Value, with 4 implied decimal places. This column is populated when the item's price was overridden at the POS and the item's original unit retail is known.	Y if unit retail was manually entered	Right/0
	Taxable Indicator	Char(1)	Refer to YSNO code_type for a list of valid codes.	Indicates whether or not item is taxable.	Y	Left/None
	Pump	Char(8)	N/A	Fuel pump identifier.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Reference Number 5	Char(30)	N/A	Number associated with a particular item within a transaction, for example, special order number. The sa_ reference table defines what this field can contain for each transaction type.	N	Left/Blank
	Reference Number 6	Char(30)	N/A	Second generic reference number at the item level.	N	Left/Blank
	Reference Number 7	Char(30)	N/A	Third generic reference number at the item level.	N	Left/Blank
	Reference Number 8	Char(30)	N/A	Fourth generic reference number at the item level.	N	Left/Blank
	Item_ swiped_ind	Char(1)	Refer to YSNO code_type for a list of valid codes	Indicates if the item was automaticall y entered into the POS system or if it had to be manually keyed.	Y	Left/None
	Return Reason Code	Char(6)	Refer to SARR code_ type for a list of valid codes.	The reason an item was returned.	N	Left/Blank
	Salesperson	Char(10)	N/A	The salesperson who sold the item.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Expiration_ date	Char(8)	N/A	Gift certificate expiration date (YYYYMMD D).	N	
	Drop Ship Ind	Char(1)	Refer to YSNO code type for a list of valid codes.	Indicates whether the item is part of a drop shipment.	Y	Left/None
	Uom_qty	Number(12)	N/A	Quantity of items purchased in the given UOM, with 4 decimal places.	Y	Right/0
	Catchweigh t_ind	Char(1)	Valid values are Y and N.	Identifies if the item is a catchweight item.		Left/None
	Selling item	Char(25)	Item identifier	Identifies the selling item.	N	Left/Blank
	Customer order line no	Number(6)	N/A	Identifies the customer order number.	N	Left/Blank
	Media id	Number(10)	N/A	Identifies the customer media ID.	N	Left/Blank
	Total Igtax Amount	Number(21)	N/A	Contains the Igtax amount.	N	Right/0
	Unique ID	Char(128)	N/A		N	Left/Blank
	Customer Order Number	Char(48)	N/A	Contains the customer order ID.	N	Left/None
	Customer Order Date	Char(14)	N/A	Contains the customer order date. Format is: YYYYMMD DHHMMSS Customer orders and layaways require customer order date.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Fulfillment Order Number	Char(48)	N/A	Contains the order ID of the fulfillment order.	N	Left/None
	No Inventory Return	Char(1)	N/A	Indicates if there is an associated inventory with the return transaction with an External Customer Order sales type.	N	Left/Blank
	Sales Type	Char(1)	N/A	Indicates if the transaction is an In Store Customer Order, External Customer Order, or Regular Sale	N	Left/Blank
	Return Warehouse	Char(10)	N/A	Contains the ID of the physical warehouse to which the inventory is returned.	N	Left/Blank
	Return Disposition	Char(10)	N/A	Contains the return disposition of the returned items.	N	Left/Blank
	Original Store	Char(10)		Contains the original store.	N	Left/Blank
	Original Transaction No	Char(10)		Contains the original transaction no.	N	Left/Blank
	Fulfillment Loc Type	Char(2)	Refer to 'FLTP' code type for a list of valid types.	Contains the fulfillment order location type. It is needed only if the file is for an OMS transaction.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Fulfillment Loc	Number(10)		Fulfillment Location ID. It is needed only if the file is for an OMS transaction.	N	Left/Blank
Item Discount	File Type Record Descriptor	Char(5)	IDISC	Identifies the file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	RMS Promotion Number	Char(6)	Refer to PRMT code_type for a list of valid types	The Merchandising promotion type.	Y	Left/Blank
	Discount Reference Number	Number(10)	N/A	Discount reference number associated with the discount type. For example, if the discount type is a promotion, this contains the promotion number.	N	Left/Blank
	Discount Type	Char(6)	Refer to SADT code_type for a list of valid types.	The type of discount within a promotion. This allows a retailer to further break down coupon discounts within the In-store promotion, for example.	N	Left/Blank
	Coupon Number	Char(40)	N/A	Number of a store coupon used as a discount.	Y if coupon	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Coupon Reference Number	Char(16)	N/A	Additional information about the coupon, usually contained in a second bar code on the coupon.	Y if coupon	Left/Blank
	Quantity Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes.	Sign of the quantity.	Y	Left/None
	Quantity	Number(12)	N/A	The quantity purchased for which the discount is applied, with 4 implied decimal places.	Y	Right/0
	Unit Discount Amount	Number(20)	N/A	Unit discount amount for this item, with 4 implied decimal places.	Y	Right/0
	Reference Number 13	Char(30)	N/A	Number associated with a particular transaction type at the discount level. The sa_ reference table defines what this field can contain for each transaction type.	N	Left/Blank
	Reference Number 14	Char(30)	N/A	Second generic reference number at the discount level.	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Reference Number 15	Char(30)	N/A	Third generic reference number at the discount level.	N	Left/Blank
	Reference Number 16	Char(30)	N/A	Fourth generic reference number at the discount level.	N	Left/Blank
	Uom_qty	Number(12)	N/A	Quantity of items purchased in the given UOM with 4 decimal places.	Y	Right/0
	Catchweight_ind	Char(1)	Valid values are Y and N.	Identifies if the item is a catchweight item.		Left/None
	Promo component	Number(10)	N/A	If the discount is a promotion, this field contains the promotion component value associated with the promotion (discount reference number).	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
Item Tax	File Type Record Descriptor	Char(5)	IGTAX	Identifies the file record type	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	Tax Authority	Char(10)	N/A	N/A	Y	Left/Blank
	Igtax Code	Char(6)	Refer to tax_ code/vat_ code of tax_ codes/vat_ codes tables.	IGtax code (tax code/VAT code) to represent whether it is a State, City, or some other tax code/VAT code.	Y	Left/Blank
	Igtax Rate	Number(20)	N/A	Igtax rate, with 4 implied decimal places.	Y	Right/0
	Igtax Amount Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes.	Sign of the Igtax amount.	Y	Left/None
	Igtax Amount	Number(21)	N/A	Total igtax amount for this item, with 5 implied decimal places.	Y	Right/0
	Reference Number 21	Char(30)	N/A	N/A	N	Left/None
	Reference Number 22	Char(30)	N/A	N/A	N	Left/None
	Reference Number 23	Char(30)	N/A	N/A	N	Left/None
	Reference Number 24	Char(30)	N/A	N/A	N	Left/None
Transaction Tax	File Type Record Descriptor	Char(5)	TTAX	Identifies the file record type.	Y	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	Tax Code	Char(6)	Refer to TAXC code_ type for as list of valid types.	Tax code (tax code/VAT code) to represent whether it is a State, City, or some other tax code/VAT code.	Y	Left/Blank
	Tax Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes	Sign of the tax amount.	Y	Left/None
	Tax Amount	Number(20)	N/A	Total Tax amount for this item, with 4 implied decimal places.	Y	Right/0
	Reference Number 17	Char(30)	N/A	N/A	N	Left/None
	Reference Number 18	Char(30)	N/A	N/A	N	Left/None
	Reference Number 19	Char(30)	N/A	N/A	N	Left/None
	Reference Number 20	Char(30)	N/A	N/A	N	Left/None
Transaction payment	File Type Record Descriptor	Char(5)	TPYMT	Identifies the file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	Payment Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes.	Sign of the deposit amount.	Y	Left/None
	Payment Amount	Number(20)	N/A	Deposit amount paid, with 4 implied decimal places.	Y	Right/0

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
Transaction Tender	File Type Record Descriptor	Char(5)	TTEND	Identifies the file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	Tender Type Group	Char(6)	Refer to TENT code_ type for as list of valid types	High-level grouping of tender types.	Y	Left/Blank
	Tender Type ID	Number(6)	Refer to the pos_tender_ type_head table for as list of valid types.	Low-level grouping of tender types.	Y	Left/Blank
	Tender Sign	Char(1)	Refer to SIGN code_ type for a list of valid codes.	Sign of the value.	Y	Left/None
	Tender Amount	Number(20)	N/A	Amount paid with this tender in the transaction, with 4 implied decimal places.	Y	Right/0

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Cc_no	Char(40)	N/A	Credit card number. Merchandise is not a PCI compliant system. Full credit card numbers are not allowed in Merchandising. The value sent in the RTLOG should be masked so that it contains no more than the first 6 digits and last 4 digits in clear text. The remaining digits should be masked using the character defined in sa_system_options.cc_no_mask_char. If more than the first 6 and last 4 characters exist in any records, the transaction file will be rejected. Alternatively, the credit card number field can be left fully blank.	N	Left/Blank
	Cc_auth_no	Char(16)	N/A	Authorization number for a credit card.	N	Left/Blank
	cc authorization source	Char(6)	Refer to CCAS code_type for as list of valid types.	N/A	N	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	cc cardholder verification	Char(6)	Refer to CCVF code_type for as list of valid types	N/A	N	Left/Blank
	cc expiration date	Char(8)	N/A	YYYYMMDD	N	Left/Blank
	cc entry mode	Char(6)	Refer to CCEM code_type for as list of valid types.	Indicates whether the credit card was swiped, thus automatically entered, or manually keyed.	N	Left/Blank
	cc terminal id	Char(5)	N/A	Terminal number from which the transaction was sent.	N	Left/Blank
	cc special condition	Char(6)	Refer to CCSC code_type for as list of valid types.	N/A	N	Left/Blank
	cc token	Char(40)	N/A	Holds unique token when the tender type used is credit, debit card, PayPal, Fonacot or Others.	N	Left/Blank
	Voucher_no	Char(25)	N/A	Gift certificate or credit voucher serial number. Voucher number needs to be included If a voucher is voided from a transaction.	Y if voucher	Right/0
	Coupon Number	Char(40)	N/A	Number of a manufacturer's coupon used as a tender.	Y if coupon	Left/Blank

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Coupon Reference Number	Char(16)	N/A	Additional information about the coupon, usually contained in a second bar code on the coupon.	Y if coupon	Left/Blank
	Cheque Account Number	Char(30)	N/A	Account number of the cheque. The value sent in the RTLOG is masked.	N	Left/Blank
	Cheque Number	Number(10)	N/A	Check number.	Required for the tender type CHECK	Right/0
	Identification Method	Char(6)	Refer to IDMH code_type for list of valid types.	Identification Method (such as a driver's license number or photo credit card).	N	Left/Blank
	Identification Id	Char(40)	N/A	Identification ID (license ID or photo card number).	N	Left/Blank
	Original Currency	Char(3)	Refer to the CURRENCIES table for valid currency codes.	The original currency with which the customer made the payment.	N	Left/Blank
	Original Currency Amount	Number(20)	N/A	Amount paid with this tender in the original currency, with 4 implied decimal places.	N	Right/0

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
	Reference No 9	Char(30)	N/A	Number associated with a particular transaction type at the tender level. The sa_ reference table defines what this field can contain for each transaction type.	N	Left/Blank
	Reference No 10	Char(30)	N/A	Second generic reference number at the tender level.	N	Left/Blank
	Reference No 11	Char(30)	N/A	Third generic reference number at the tender level.	N	Left/Blank
	Reference No 12	Char(30)	N/A	Fourth generic reference number at the tender level.	N	Left/Blank
Transaction Trailer	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	Transaction Record Counter	Number(10)	N/A	Number of records processed in the current transaction (only those records between transaction head and tail).	N/A	N/A

Table 27–33 (Cont.) File Name: rtlog

Record Name	Field Name	Field Type	Default Value	Description	Required?	Justification /Padding
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies the file record type.	Y	Left/Blank
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Y	Right/0
	File Record Counter	Number(10)	N/A	Number of transactions processed in the current file (only the records between the file head and tail).	Y	Right/0

The RTLOG file is imported into the Sales Audit tables after validation by the batch program saimptlog. This section describes the requirements and validations performed on the records.

Common Requirements/Validations

This section details the common requirements and validations performed on all transactions. The following sections describe the specific requirements of each type of transaction. If a transaction is not mentioned, it does not have specific requirements.

Table 27–34 Common Requirements and Validations

Transaction Type	Includes item records?	Includes tender records?	Includes tax records? IG TAX?	Includes customer records?
OPEN	No	No	No	No
NOSALE	No	Optional	No	No
VOID	Optional	Optional	Optional	Optional
PVOID	No	No	No	No
SALE	Optional	Yes	Optional	Optional
RETURN	Yes	Yes	Optional	Optional
EEXCH	Yes	No	Optional	Optional
PAIDIN	No	Yes	No	No
PAIDOU	No	Yes	No	No
PULL	No	Yes	No	No
LOAN	No	Yes	No	No
COND	No	No	No	No
CLOSE	No	No	No	No
TOTAL	No	No	No	No

Table 27–34 (Cont.) Common Requirements and Validations

Transaction Type	Includes item records?	Includes tender records?	Includes tax records? IG TAX?	Includes customer records?
REFUND	This transaction is not sent through the RTLOG. It is entered at the HQ level. The TITEM and TCUST records are optional. The TTEND record is required. A TTAX record should not be included if IGTAX appears in a transaction. IGTAX is an item-level tax and TTAX is a transaction-level tax. Either IGTAX or TTAX can be used, but not both.			
METER	Yes	No	No	No
PUMPT	Yes	No	No	No
TANKDP	Yes	No	No	No
TERM	TERM records are created by saimptlog and then loaded into the database. They do not come from the RTLOG file. They require one TITEM, one TTEND, one TTAX, one TCUST record, and one CATT record, IGTAX, and one TPYMT which is newly coming up.			
DCLOSE	No	No	No	No
SPLORD	Optional	Yes	Optional	Optional
REOPEN	No	No	No	No

Requirements per Record Types

Table 27–35 Requirements per Record Type

Record Type	Requirements
IDISC	IDISC records must immediately follow their associated TITEM record.
IGTAX	IGTAX will immediately follow TITEM if IDISC is not coming, otherwise it should follow IDISC. Even if IGTAX is coming prior to IDISC, it will be processed, but for maintaining proper format, Sales Audit expects it to come after IDISC.
TTAX	Either this record or IGTAX should appear in the transaction. IGTAX and TTAX cannot be both used at the same time.
TPYMT	This record should be right before the TTEND record. It contains the deposit amount for pickup/delivery/layaway orders.
CATT	CATT records must immediately follow their associated TCUST record.

Code Type Validations

Table 27–36 Code Type Validations

Record Name	Field Name	Code Type
Transaction Header	Transaction Type	TRAT
	Sub-transaction Type	TRAS
	Reason Code	REAC or values from non_merch_code_ head if the transaction type is PAIDOU and the sub-transaction type is MV or EV.

Table 27–36 (Cont.) Code Type Validations

Record Name	Field Name	Code Type
	Value Sign	SIGN
	Vender No	If the transaction type is PAIDOU and the sub-transaction type is MV, this field is validated against the supplier table. If the transaction type is PAIDOU and the sub-transaction type is EV, this field is validated against the partner table.
	Transaction Processing System	TSYS
Transaction Item	Item Type	SAIT
	Item Status	SASI
	Item Number Type	UPCT
	Quantity Sign	SIGN
	Taxable Indicator	YSNO
	Price Override Reason Code	ORRC
	Item Swiped Indicator	YSNO
	Sales Type	SASY
	Return Disposition	INV_STATUS_CODES table
	No Inventory Return	YSNO
	Return Reason Code	SARR
	Fulfillment Loc Type	FLTP
Item Discount	RMS Promotion Type	PRMT
	Discount Type	SADT
	Quantity Sign	SIGN
Transaction Customer	Customer ID Type	CIDT
Customer Attribute	Attribute Type	SACA
	Attribute value	Code types from the codes in SACA.
Transaction Tax	Tax code	TAXC from the CODE_DETAIL table or VATC from the VAT_CODES table.
	Tax sign	SIGN
Transaction Payment	Payment (Deposit Amount) Sign	SIGN

Table 27–36 (Cont.) Code Type Validations

Record Name	Field Name	Code Type
Transaction Tender	Transaction Tender Tender Type Group	TENT
	Tender Sign	SIGN
	Tender Type ID	Pos_tender_type_head table
	CC Authorization Source	CCAS
	CC Cardholder Verification	CCVF
	CC Entry Mode	CCEM
	CC Special Condition	CCSC

The following dates are validated: Business Date, Transaction Date, and Expiration Date. Also, saimptlog accepts only business dates that are within the PERIOD.VDATE minus the SA_SYSTEM_OPTIONS.DAYS_POST_SALE value.

The store number is validated against the STORE table. Numeric fields are checked for non-numeric characters.

For transactions of type SALE, RETURN, and EEXCH, saimptlog checks whether a transaction is in balance. With the introduction of the Item level tax and Payment amount lines, the balancing logic has been changed as below. Also with introduction of handling VAT/TAX, the logic of balancing has been modified as below.

- When TAX is on in the system (system_options.default_tax_type equals SALES):

  Transaction Items (Unit Retail * Unit Retail Sign * Quantity) of items which are on Regular Sale, Return, or EEXCH

  + Item Discounts (Unit Discount Amount * Unit Discount Sign * Quantity) of items which are on Regular Sale, Return, or EEXCH

  + Item Level Tax (Total Igtax Amount) of items which are on Regular Sale, Return, or EEXCH

  + Transaction Tax (Tax Amount * Tax Sign)

  + Transaction payment (Payment Amount * Payment Sign)

  equals Transaction Tenders (Tender Amount * Tender Sign)

  saimptlog will populate the Value field (on THEAD) with the transaction's sales value (item value minus discount value plus tax value) from the preceding calculation if it was not provided in the RTLOG. The following change is made in the sale total balancing: Value field in THEAD will be: (item value - discount value + tax value) for items which are on Regular Sale, Return, or EEXCH + payment value.

  Note: If this Value field is being used in creating some totals, then accordingly, these totals needs to be modified to accommodate the extra amount coming in.

- When VAT is on in the system (system_options.default_tax_type in GTAX, SVAT), look for other system options, along with class level and store level VAT

indicators, which tell whether the unit retail is inclusive or exclusive of VAT. The logic of balancing will vary:

Transaction Items (Unit Retail * Unit Retail Sign * Quantity) of items which are on Regular Sale, Return, or EEXCH

+ Item Discounts (Unit Discount Amount * Unit Discount Sign * Quantity) of items which are on Regular Sale, Return, or EEXCH

+ Item Level Tax (Total Igtax Amount) of items which are on Regular Sale, Return, or EEXCH (when VAT is off at the item level).

+ Transaction Tax (Tax Amount * Tax Sign)

+ Transaction Payment (Payment Amount * Payment Sign)

equals Transaction Tenders (Tender Amount * Tender Sign)

Vouchers are treated as follows:

- If an item sold is as a gift certificate (Transaction Item, Voucher field has a value), the issued information is written to the SA_VOUCHER table.
- If the Transaction Type is RETURN, and the Transaction Tender Type Group is voucher (VOUCH), the issued information is written to the SA_VOUCHER table.
- If the Transaction Type is SALE and the Transaction Tender Type Group is a voucher (VOUCH), the redeemed information is written to the SA_VOUCHER table.
- When a gift certificate is sold, the customer information should always be included. A receiving customer name value should be populated in the ref_no5 field, receiving customer state value should be populated in the ref_no6 field, and receiving customer country should be populated in the ref_no7 field. These reference fields can be changed by updating the sa_reference table, but the code needs to be modified as well. The expiration date is put in the expiration_date field in the TITEM record.

Other validations and points to consider:

- The salesperson in the TITEM record takes precedence over the salesperson in the THEAD record.
- If an item sold is a sub-transaction (REF) item (Transaction Item, reference item field has a value and item does not), it will be converted to the corresponding transaction level item (ITEM).
- If an item sold is an ITEM (Transaction Item, item field has a value), it will be validated against the Merchandising item tables.
- The corresponding Department, Class, Subclass, and Taxable Indicator will be selected from the Merchandising tables and populated for an item.

The balancing level determines whether the register or the cashier fields are required:

- If the balancing level is R (register), the register field on the THEAD must be populated.
- If the balancing level is C (cashier), the cashier field on the THEAD must be populated.
- If the balancing level is S (store), neither field is required to be populated.
- The tax_ind and the item_swiped_ind fields can only accept Y or N values. If an invalid value is passed through the RTLOG, an error will be flagged and the value will be defaulted to Y.

Transaction of Type SALE

A transaction of type SALE is generated whenever an item is sold. If a sale is for an employee, the sub-transaction type is EMP. If it is a drive-off sale, when someone drives off with unpaid gas, the sub-transaction type is DRIVEO. A special type of sale is an odd exchange, sub-transaction type EXCH, where items are sold and returned in the same transaction. If the net value of the exchange is positive, then it is a sale. If the net value is negative, it is a return.

Requirements per record type (other than what is described in the preceding Layout section):

Table 27–37 Requirements per Record Type

Record Type	Requirements
THEAD	N/A
TITEM	■ Item Status is a required field; it determines whether the item is Sold (S), Returned (R), or Voided (V). If the item status is S, the quantity sign is expected to be P. If the item status is R, the quantity sign is expected to be N. Also, if the item status is ORI, LIN, ORD, or LCO, the quantity sign should be P. In the case of ORC or LCA, it should be N. ■ If the item status is V, the quantity sign is the reverse of the quantity sign of the voided item. That is, if an item with status S is voided, the quantity sign would be N. Furthermore, the sum of the quantities being voided cannot exceed the sum of the quantities that are Sold or Returned. Note: Neither of the two validations are performed by saimptlog, but an audit rule could be created to check this. ■ The following item statuses are used for handling items on customer order layaway: ORI - Order Initiate ORD - Order Complete ORC - Order Cancel LIN - Layaway Initiate LCA - Layaway Cancel LCO - Layaway Complete ■ In a typical sale, the items all have a status of S. In the case of an odd exchange, some items will have a status of R. ■ In a typical return, the items all have a status of R. In the case of an odd exchange, some items will have a status of S. ■ If an item has status R, then the Return Reason Code field may be populated. If it is, it will be validated against code type SARR. Also, it is better to capture the Return Reason Code in the case of items on ORC or LCA, but it is not mandatory. No validation is kept for these new item statuses for checking of SARR. ■ If the price of an item is overridden, the Override Reason and Original Unit Retail fields must be populated.

Table 27–37 (Cont.) Requirements per Record Type

Record Type	Requirements
IDISC	■ The Merchandising Promotion Type field must always be populated with values of code type PRMT. ■ The Promotion field is validated, when a value is passed, against the promhead table. ■ If the promotion is In Store (code 1004), the Discount Type field must be populated with values of code type SADT. ■ The Discount Reference Number is a promotion number which is of status A, E, or M. ■ If the Discount Type is SCOUP for Store Coupon, the Coupon Number field must be populated. The Coupon Reference Number field is optional.
IGTAX	■ The IGTAX_CODE field must always be populated depending on the system's default tax type. For a default tax type of SALES, this field will be populated with values of code_type TAXC. For a default tax type of SVAT or GTAX,, this field will be populated with VATC (vat_code from vat_codes table). IGTAX is an Item-level tax. ■ The TAX_AUTHORITY field must always be populated when the default tax type is GTAX.
TTAX	■ The TAX_CODE field must always be populated depending on the system's default tax type. For a default tax type of SALES, this field will be populated with values of code_type TAXC. For a default tax type of GTAX or SVAT, this field will be populated with VATC (vat_code from vat_codes table). TTAX is a Transaction-level tax.
TPYMT	Payment (Deposit amount) sign and Payment (Deposit) amount fields are necessary if this line is appearing. Basically, this is the accumulation of various items being considered in one transaction, which are on pick up/delivery/lay away.
TTEND	If the tender type group is COUPON, the Coupon Number field must be populated. The Coupon Reference Number field is optional.

Meaning of reference number fields:

Note: The meaning of these reference number fields may be changed through the sa_reference table. The transaction type SPLORD is the same as SALE, but the inventory will not be reserved for the orders at its line level.

Table 27–38 Meaning of Reference Number Fields

Transaction Type	Sub-transaction Type	Item Type	Tender Type Group	Reference Number Field	Meaning of Reference Field	Req?
SALE	N/A	N/A	N/A	1	Speed Sale Number	Y
SALE	N/A	GCN	N/A	5	Recipient Name	N
SALE	N/A	GCN	N/A	6	Recipient State	N
SALE	N/A	GCN	N/A	7	Recipient Country	N
SALE	N/A	N/A	CHECK	9	Check Number	N
SALE	N/A	N/A	CHECK	10	Driver's License Number	N
SALE	N/A	N/A	CHECK	11	Credit Card Number	N
SALE	DRIVEO	N/A	N/A	1	Incident Number	Y
SALE	EMP	N/A	N/A	3	Employee Number of the employee receiving the goods.	N

Table 27–39 Expected Values for Sign Fields

TRANSACTION TYPE	TITEM.Quantity Sign	TEND.Tender Sign	TTAX.Tax Sign	IDISC.Quantity Sign
SALE	P if item is sold; N if item is returned; reverse of original item if item is voided.	P	P	P if item is sold; N if item is returned; reverse of original item if item is voided.
SALE	P if item is on ORI, LIN, ORD, or LCO; N if item is on ORC or LCA.	P	P	P if item is on ORI, LIN, ORD, or LCO; N if item is on ORC or LCA.

Transaction of Type PVOID

This transaction is generated at the register when another transaction is being post voided. The orig_tran_no and orig_reg_no fields must be populated with the appropriate information for the transaction being post voided. The PVOID transaction must be associated with the same store day as the original transaction. If the PVOID needs to be generated after the store day is closed, the transaction needs to be created using the forms.

Transaction of type RETURN

This transaction is generated when a customer returns an item.

This type of transaction has similar record type requirements as a SALE transaction.

Meaning of reference number fields:

> **Note:** The assumption is that new item statuses will not come under transaction type RETURN.

If a customer wants to return the items (ORI, LIN), these will come under SALE but with item statuses as ORC or LCA.

> **Note:** The meaning of these reference number fields may be changed through the sa_reference table.

Table 27–40 Meaning of Reference Number Fields

Transaction Type	Sub-transaction Type	Reference Number Field	Meaning of Reference Field	Req?
RETURN	N/A	1	Receipt Indicator (Y/N)	Y
RETURN	N/A	2	Refund Reference Number	N
RETURN	EMP	3	Employee Number of the employee returning the goods.	N

Table 27–41 Expected Values for Sign Fields

TRANSACTION TYPE	TITEM.Quantity Sign	TEND.Tender Sign	TTAX.Tax Sign	IDISC.Quantity Sign
RETURN	P if item is sold; N if item is returned; reverse of original item if item is voided.	N	N	P if item is sold; N if item is returned; reverse of original item if item is voided

Transaction of type SPLORD

This transaction is generated when a customer picks up an item, which is not in stock. The item status can be ORI, ORC, or ORD. (Order Initiate, Order Cancel, or Order Complete).

Transaction of type EEXCH

This transaction is generated when there is an even exchange.

This type of transaction has similar record type requirements as a SALE transaction.

It is expected that the number of items returned equals the number of items sold. However, this validation is not performed by saimptlog. An audit rule could be created for this. Saimptlog only expects that there would be at least two item records.

No tender changes hands in this transaction.

Meaning of reference number fields:

Note: The items, which are on customer order or layaway, should not be come under this transaction type.

The meaning of these reference number fields may be changed through the sa_reference table.

Table 27–42 Meaning of Reference Number Fields

Transaction Type	Sub-transaction Type	Reference Number Field	Meaning of Reference Field	Req?
EEXCH	N/A	1	Receipt Indicator (Y/N)	Y
EEXCH	EMP	3	Employee Number of the employee exchanging the goods.	N

Transaction of type PAIDIN

This type of transaction has only one TTEND record.

A reason code is required.

Meaning of reference number fields:

Note: The meaning of these reference number fields may be changed through the sa_reference table.

Table 27–43 Meaning of Reference Number Fields

Reason Code	Reference Number Column	Meaning	Req?
NSF	1	NFS Check Credit Number	N
ACCT	1	Account Number	N

Transaction Type PAIDOU

This type of transaction has only one TTEND record.

A reason code is required (code type REAC). If the sub-transaction type is EV or MV, the reason code comes from the non_merch_codes_head table.

If the sub-transaction type is EV or MV, then at least one field among the vendor number, vendor invoice number, payment reference number, and proof of delivery number fields should be populated.

If the sub-transaction type is EV, the vendor number comes from the partner table. If the sub-transaction type is MV, the vendor number comes from the supplier table.

Meaning of reference number fields:

Note: The meaning of these reference number fields may be changed through the sa_reference table.

Table 27–44 Meaning of Reference Number Fields

Sub Transaction Type	Reason Code	Reference Number Column	Meaning	Req?
EV	N/A	2	Personal ID Number	N
EV	N/A	3	Routing Number	N
EV	N/A	4	Account Number	N
NA	PAYRL	1	Money Order Number	N
NA	PAYRL	2	Employee Number	N
NA	INC	1	Incident Number	N

Transaction of Type PULL

This transaction is generated when cash is withdrawn from the register.

This type of transaction has only one TTEND record.

Expected values for sign fields

Table 27–45 Expected Values for Sign Fields

TRANSACTION TYPE	TITEM.Quantity Sign	TEND.Tender Sign	TTAX.Tax Sign	IDISC.Quantity Sign
PULL	N/A	N	N/A	N/A

Transaction of Type LOAN

This transaction is generated when cash is added to the register.

This type of transaction has only one TTEND record.

Expected values for sign fields:

Table 27–46 Expected Values for Sign Fields

TRANSACTION TYPE	TITEM.Quantity Sign	TEND.Tender Sign	TTAX.Tax Sign	IDISC.Quantity Sign
LOAN	N/A	P	N/A	N/A

Transaction Type Cond

This transaction records the condition at the store when it opens. There can be at most one COND record containing weather information and at most one COND record containing temperature information. Both of these pieces of information may be in the same COND record. There may be any number of COND records containing traffic and construction information.

This type of transaction does not have TITEM, IDISC, IGTAX, TTAX, TPYMT, or TTEND records

Note: The meaning of these reference number fields may be changed through the sa_reference table.

Table 27–47 Meaning of Reference Number Fields

Reference Number Column	Meaning	Req?
1	Weather - code type WEAT	N
2	Temperature - a signed 3 digit number.	N
3	Traffic - code_type TRAF	N
4	Construction - code_type CONS	N

Transaction of Type TOTAL

This transaction records the totals that are reported by the POS and OMS. The value field must be populated. Some systems generate only one transaction number for all totals. In order to avoid duplicate errors being reported, only one total transaction can have a transaction number and the subsequent ones can have blank transaction numbers. In other words, a TOTAL transaction is not required to have a transaction number.

This type of transaction does not have TITEM, IDISC, IGTAX, TTAX, TPYMT, TTEND records.

Transaction of Type METER

This transaction is generated when a meter reading of a fuel pump is taken.

This type of transaction has only TITEM records.

Meaning of reference number fields:

Note: The meaning of these reference number fields may be changed through the sa_reference table.

Table 27–48 Meaning of Reference Number Fields

Reference Number Column	Meaning	Req?
1	Reading Type: (A for adjustment, S for shift change, P for price change, or C for store close)	Y
5	Opening Meter Readings	Y
6	Closing Meter Reading	Y
7	If the reading type is P for price change, the old unit retail should be placed here. Decimal places are required.	Y
8	Closing Meter Value	Y

Transaction of Type PUMPT

This transaction is generated when a pump test is performed. This type of transaction has only TITEM records.

Transactions of Type TANKDP

This transaction is generated when a tank dip measurement is taken.

This type of transaction has only TITEM records.

Meaning of reference number fields:

Note: The meaning of these reference number fields may be changed through the sa_reference table.

Table 27–49 Meaning of Reference Number Fields

Reference Number Column	Meaning of Reference Field	Req?
1	Tank identifier	Y
5	Dip Type (FUEL, WATER, and so on)	Y
6	Dip Height Major (decimal places required)	Y
7	Dip Height Minor (decimal places required)	Y

Transaction of Type DCLOSE

This transaction is generated when the day closed. The transaction number for this type of transaction has to be blank.

Note: Vouchers are minimally handled by saimptlog. Voucher information is written to the savouch file which is passed to the program savouch.pc.

- A voucher will appear on the TITEM record only if it was sold. When saimptlog encounters a SALE transaction with a voucher, it writes the voucher to the savouch file as an I for Issued voucher.
- A voucher will be issued when it appears on the TTEND record of transactions of type RETURN and PAIDOU. In other words, saimptlog will write it to the savouch file with status I.
- A voucher will be redeemed when it appears on the TTEND record of transactions of type SALE and PAIDIN. In other words, saimptlog will write it to the savouch file with status R.

Vouchers may not be returned. However, a transaction of type PAIDOU may be generated when the customer exchanges a voucher for another form of tender.

Transaction of Type REOPEN

This transaction is generated when a store day which was closed needs to be reopened to process additional transactions. Transaction number for this type of transaction has to be blank.

Transaction of Type OTHER

This transaction is a generic transaction type to support Micros Xstore integration. This will identify all the other transaction types that are not currently supported. This type of transaction has only THEAD and TTAIL records.

Design Assumptions

Table 27–50 Sales Audit Valid Transaction Type

Transaction Code	Transaction Type
OPEN	Open
CLOSE	Close
COND	Daily Store Conditions
DCLOSE	Day close indicator
LOAN	Loan
METER	Meter Reading for Fuel
NOSALE	No Sale
PAIDIN	Paid In
PAIDOU	Paid Out
PULL	Pull
PUMPT	Pump Test for Fuel
PVOID	Post Void (A transaction that was rung later into the register to void something that occurred earlier at the same store/day. A post void updates the original transaction's sub-transaction type.)
REFUND	Return of customer's original check.
RETURN	Return
SALE	Sale
TANKDP	Tank Dip

Table 27–50 (Cont.) Sales Audit Valid Transaction Type

Transaction Code	Transaction Type
TOTAL	POS generated totals
EEXCH	Even exchange
VOID	Void (aborted transaction)
OTHER	Others
REOPEN	Reopen Store Day from POS

DCLOSE Transaction Type

When the retailer is sending only one file to the system, SAIMPTLOG.PC marks the store day record in the Sales Audit import log as partially or fully loaded in the database by looking for a transaction type of DCLOSE. However, if the retailer is sending more than one file (as in, for example, a trickle polling situation), the retailer can specify the number of files that the system should expect in combination with the DCLOSE transaction type. This ensures that the system receives all of the files, even if the DCLOSE transaction type is, for some reason, received before the final file.

For example, if 24 files are expected over a given amount of time, and the file with the DCLOSE transaction type is, for some reason, sent before the 24th file, the Merchandising system waits until the last file arrives before marking the store day record as partially or fully loaded in the database.

The import process is completed after SAIMPTLOGFIN.PC has updated the store, data, and audit status of each store day record.

The Reopen Transaction Type

When the retailer is sending transaction of type of REOPEN for store and business day system should expect REOPEN as first transaction in the file before any additional transactions.

When secondary DCLOSE transaction is sent after REOPEN transaction type system should expect count of files since the prior DLCOSE transaction (not the full count for store/day).

SAIMPTLOGFIN.PC batch program would sum up the file counts in case of multiple DCLOSE transactions for the store day and compare against the files loaded in Sales Audit and update the store, data and audit status.

saimptlogtdup_upd (Processing to Allow Re-Upload of Deleted Transactions)

Module Name	saimptlogtdup_upd.pc
Description	Processing to Allow Re-Upload of Deleted Transactions
Functional Area	Oracle Retail Sales Audit
Module Type	Admin
Module Technology	ProC
Catalog ID	RSA19

Wrapper Script batch_saimptlogtdup_upd.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this batch module is to fetch all deleted transactions for a store day and modify the tdup<Store><rtlog originating system>.dat file to remove deleted transactions, from the tdup range, in order to facilitate the saimptlog/saimptlogi batch to upload deleted transactions again. The batch will process all the store day with data status in Partially Loaded and Ready For Import and a business date that lies between the vdate minus the sa_syatem_options. day_post_sale and the vdate. The batch will not process a store day, if the tdup<Store><rtlog originating system>.dat file does not exist. The batch is designed to work only if sa_system_options.check_dup_miss_tran is set to Y, otherwise, do nothing and come out with successful completion. Also, the batch will not terminate with an error, if the deleted transaction to be removed from tdup range does not exist in the tdup<Store><rtlog originating system>.dat file.

Restart/Recovery

N/A

Design Assumptions

N/A

saimptlogfin (Complete Transaction Import Processing)

Module Name	saimptlogfin.pc
Description	Complete Transaction Import Processing
Functional Area	Oracle Retail Sales Audit
Module Type	Admin
Module Technology	ProC
Catalog ID	RSA38
Wrapper Script	batch_saimptlogfin.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The saimptlogfin program creates the balances (over or under) by store, register, or cashier and populates it in the SA_BALANCE_GROUP table. It also cancels post voided transactions and vouchers and validates missing transactions. It marks the store day record in the Sales Audit import log as partially or fully loaded. This will unlock the store day records after all store transactions are imported.

Restart/Recovery

N/A

Design Assumptions

N/A

savouch (Sales Audit Voucher Upload)

Module Name	savouch.pc
Description	Sales Audit Voucher Upload
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA08
Wrapper Script	batch_savouch.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Because gift certificates can enter the Sales Audit system as either items or tender, processing must be done to match up the sales and redemptions. This module is used to aggregate gift certificate and voucher records. It compares records in the input files to the database. If a record for the voucher does not exist on the database, the record is inserted. If the voucher already exists on the database, the record should be updated with the appropriate information. The voucher details are updated to SA_VOUCHER table.

Some retailers assign gift certificates to a given store, which means that before a gift certificate is sold at a store, it is assigned to a given store. When a retailer assigns a gift certificate to a given store, a record is written to the database. When the gift certificate is then sold by the store and redeemed by the consumer, this existing record must be updated to include the sale and redemption information. Some retailers choose not to assign gift certificates and instead simply sell gift certificates. In that case, the record will be inserted into the database when the gift certificate is sold and then updated when the gift certificate is redeemed.

Restart/Recovery

Restart/recovery logic for file-based processing is used. Records will be committed to the database when the commit_max_ctr defined in the RESTART_CONTROL table is reached.

I/O Specification

Integration Type Upload to Sales Audit

File Name The input file name is not fixed; the input file name is determined by a runtime parameter. Records rejected by the import process are written to a reject file. The reject file name is not fixed; the reject file name is determined by a runtime parameter.

Integration Contract IntCon000160 (SAVO)

Input File Layout

Table 27–51 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor/	Char(5)	FHEAD	File head marker.
	Line id	Number(10)	0000000001	Unique line ID.
	Translator id	Char(5)	SAVO	Identifies transaction type.
	File create date	Char(14)	N/A	Vdate in YYYYMMDDHH24MISS format.
	Business Date	Char(8)	Business Date	Vdate in YYYYMMDD format.

Table 27–51 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FDETL	Record descriptor/	Char(5)	FDETL	File head marker.
	Line id	Number(10)	N/A	Unique line ID.
	Voucher seq Number	Number(20)	N/A	Unique identifier for an entry to the SA_VOUCHER table.
	Voucher No	Char(16)	N/A	Serial Number of the voucher.
	Tender Type Id	Number(6)	N/A	Type of Voucher (Valid values for tender type are maintained in the pos_ tender_type_head table with tender_type_group as VOUCH.
	Assigned Date	Char(8)	N/A	Date the voucher was assigned.
	Assigned store	Number(10)	N/A	Store to which the voucher is assigned.
	Issuing Date	Char(8)	N/A	Date this document was issued.
	Issuing store	Number(10)	N/A	Store this document was issued from.
	Issuing Register	Char(5)	N/A	Register this document was issued from.
	Issuing Cashier	Char(10)	N/A	Cashier issuing the document.
	Issued transaction number	Number(20)	N/A	Transaction number at the time of issuance.
	Issued item seq number	Number(4)	N/A	Will hold the item sequence of the item when the voucher is sold as an item (gift voucher).

Table 27–51 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Issued tender seq number	Number(4)	N/A	Will hold the tender sequence of the tender when the voucher is sold as a tender (Merchandise Credit).
	Issued Amount	Number(20)	N/A	Amount the voucher was issued for*10000 (4 implied decimal places).
	Issued Customer Name	Char(120)	N/A	Name of the customer, who was issued the voucher.
	Issued Customer Addr1	Char(240)	N/A	The address of the customer who was issued the voucher.
	Issued Customer Addr2	Char(240)	N/A	The second line address of the customer who was issued the voucher.
	Issued Customer City	Char(120)	N/A	City of the customer, the voucher is issued.
	Issued Customer State	Char(3)	N/A	State of the customer.
	Issued Customer Postal Code	Char(30)	N/A	Postal address of the customer.
	Issued Customer Country	Char(3)	N/A	Country of the customer where the voucher was issued.
	Recipient Name	Char(120)	N/A	Name of the intended recipient.
	Recipient State	Char(3)	N/A	The state of the intended recipient.
	Recipient Country	Char(3)	N/A	The country of the intended recipient.
	Redemption Date	Char(8)	N/A	Date the voucher was redeemed.
	Redemption Store	Number(10)	N/A	Store at which the voucher was redeemed.
	Redemption Register	Char(5)	N/A	Register at which the document was redeemed.
	Redemption cashier	Char(10)	N/A	Cashier redeeming the voucher.
	Redemption tran seq number	Number(20)	N/A	Transaction Number when the document was redeemed.

Table 27–51 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Redemption Tender seq number	Number(4)	N/A	This column will hold the tender sequence of the tender within the transaction when a voucher is redeemed as tender.
	Redemption Amount	Number(20)	N/A	Amount the voucher was redeemed for*10000 (4 implied decimal places).
	Expiry Date	Char(8)	N/A	Expiry Date.
	Status	Char(1)	N/A	ndicator showing the document's status - issued or redeemed. Valid values = I - Issued, R - Redeemed.
	Comments	Char(2000)	N/A	Comments.
FTAIL	Record type	Char(5)	FTAIL	Describes file record and marks the end of file.
	Line id	Number(10)	N/A	Unique file line ID.
	#lines	Number(10)	N/A	Total number of transaction lines in file (not including FHEAD and FTAIL).

Design Assumptions

N/A

saimpadj (Import Total Value Adjustments From External Systems to ReSA)

Module Name	saimpadj.pc
Description	Import Total Value Adjustments From External Systems to Sales Audit
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA07
Wrapper Script	rmswrap_in_rej.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module posts external system adjustments to the Sales Audit total value table.

The sales audit adjustments are passed to the module in an external file.

Records that fail necessary validations would be written to the reject file. The input and reject file names are passed as arguments.

Restart/Recovery

Restart/recovery logic for file-based processing is used. The logical unit of work for this module is a parameterized number defined in the restart tables.

Record level locking is done on sa_store_day before updating.

I/O Specification

Integration Type Upload to Sales Audit

File Name Determined by runtime parameters.

Integration Contract IntCon000047

Input File Layout

Table 27–52 Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type (the beginning of the input file).
	File Line Identifier	Number(10)	Sequential number	ID of the current line being read from input file.
	File head descriptor	Char(4)	IMPA	Describes file line type.
	Current date	Char(14)	N/A	File date in YYYYMMDDHH24MISS format.

Table 27–52 (Cont.) Input File Layout

Record Name	Field Name	Field Type	Default Value	Description
FDETL	File Type Record Descriptor	Char(5)	FDETL	Identifies the file record type to upload a new deal header.
	File Line Identifier	Number(10)	Sequential number	ID of the current line being read from input file.
	Data source	Char(6)	N/A	Name of the external system that produced the file.
	New value sign	Char(1)	N/A	Sign(+/-) for the new value.
	New Value	Number(20)	N/A	Value for the total entered by Headquarters user*10000 (4 implied decimal places).
	Total seq no	Number(20)	N/A	Identifies the unique result set for this total ID, total revision, or store/day. Balancing group and index values.
	Store	Number(10)	N/A	Store number for a store/day combination.
	Business Date	Char(8)	N/A	Date for store/day combination.
	Total id	Char(10)	N/A	ID to uniquely identify the total.
	Ref no 1	Char(30)	N/A	The first reference value based by which the total is grouped.
	Ref no 2	Char(30)	N/A	The second reference value based by which the total is grouped.
	Ref no 3	Char(30)	N/A	The third reference value based by which the total is grouped.
FTAIL	File Type record descriptor	Char(5)	FTAIL	Identifies the file record type (the end of the input file).
	File Line Identifier	Number(10)	Sequential number	ID of the current line being read from input file.
	File Record Counter	Number(10)	Sequential number	Number of records/transactions in the current file (only records between head and tail).

Design Assumptions

N/A

satotals (Calculate Totals based on Client Defined Rules)

Module Name	satotals.pc
Description	Calculate Totals based on Client Defined Rules
Functional Area	Sales Audit, Totals
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RSA16
Wrapper Script	batch_satotals.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module produces totals from user-defined total calculation rules. Totaling is integral to the sales auditing process. Totaling provides the values against which auditors can compare receipts. These comparisons find data errors that could be the result of either honest mistakes or fraud. Finding these mistakes during the sales auditing process prevents these errors from being passed on to merchandising and data warehouse systems. Totaling also provides quick access to other numeric figures about the day's sales transactions.

Totaling in Sales Audit is dynamic. Sales Audit automatically totals transactions based on calculation definitions that the retailer's users create using the online Totals Calculation Definition Wizard. In addition, the retailer is able to define totals that come from the POS, but that Sales Audit does not calculate. Whenever you create new calculation definitions or edit existing ones, they become part of the automated totaling process the next time that this process runs.

Restart/Recovery

The logical unit of work for this program is a SA_STORE_DAY record. Records are committed to the database when the commit_max_ctr defined for SATOTALS on the RESTART_CONTROL table is reached. This program achieves inherent restart/recovery due to the fact that store/day records that are processed will be updated to an audit_status of T for Totaled and will not be fetched by the driving cursor when the program restarts.

Design Assumptions

N/A

sa_totals_calc_job (Calculate Totals based on Client Defined Rules)

Module Name	sa_totals_calc_job
Description	Calculate Totals based on Client Defined Rules
Functional Area	Sales Audit, Totals

Module Type	Admin - Ad hoc
Module Technology	Background Processing
Integration Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (SA_TOTALS_CALC_THREAD) will filter eligible records from sales audit store day (sa_store_day) table for all stores wherein auditing status is "R"e-Totaling/Auditing Required. Totaling provides the values against which auditors can compare receipts. These comparisons find data errors that could be the result of either honest mistakes or fraud. Finding these mistakes during the sales auditing process prevents these errors from being passed on to merchandising and data warehouse systems. Totaling also provides quick access to other numeric figures about the day's sales transactions.

Totaling in ReSA is dynamic. ReSA automatically totals transactions based on calculation definitions that the retailer's users create using the online Totals Calculation Definition Wizard. In addition, the retailer is able to define totals that come from the POS but that ReSA does not calculate. Whenever users create new calculation definitions or edit existing ones, they become part of the automated totaling process the next time that this program runs. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_SA_TOTALS_CALC_STG.

The Business logic program (SA_TOTALS_CALC) will process all records from the staging table. Using bulk processing, this program will process the records for totals build-up and calculation by calling SA_BUILD_TOTAL_SQL.PROCESS_CALC_TOTALS for each store day captured. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 27–53 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 times a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Store Day Sequence number

Restart/Recovery

NA

Key Tables Affected

Table 27–54 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_SA_TOTALS_CALC_STG	Yes	Yes	No	Yes
SA_STORE_DAY	Yes	No	Yes	No
SA_TOTAL	No	Yes	No	No
SA_TOTAL_HEAD	Yes	No	No	No
SA_ERROR	No	Yes	No	Yes
SA_ERROR_WKSHT	No	Yes	No	Yes
SA_POS_VALUE	No	Yes	No	No
SA_POS_VALUE_WKSHT	No	Yes	No	No
SA_SYS_VALUE	No	Yes	No	No
SA_SYS_VALUE_WKSHT	No	Yes	No	No
SA_ERROR_REV	No	Yes	No	No
SA_EXPORTED_REV	No	Yes	No	No
SA_EXPORTED	No	No	No	Yes

Design Assumptions

NA

sarules (Evaluate Transactions and Totals based on Client Defined Rules)

Module Name	sarules.pc
Description	Evaluate Transactions and Totals based on Client Defined Rules
Functional Area	Oracle Retail Sales Audit
Module Type	Business Processing
Module Technology	ProC
Catalog ID	RSA17
Wrapper Script	batch_sarules.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

Evaluating rules is integral to the sales auditing process. Rules make the comparisons between data from various sources. These comparisons find data errors that could be the result of either honest mistakes or fraud. Finding these mistakes during the sales auditing process prevents these errors from being passed on to merchandising and data warehouse systems.

Rules in Sales Audit are dynamic. Aside from basic data validations, rules are not predefined in the system. Retailers have the ability to define them through the online Rule Definition Wizard. Errors uncovered by these rules are available for review online during the interactive audit process. After you modify existing rules or create new ones, they become part of the rules the next time that sarules.pc runs.

Restart/Recovery

The logical unit of work for this program is a SA_STORE_DAY record. Records are committed to the database when the commit_max_ctr defined for SARULES on the RESTART_CONTROL table is reached. This program achieves inherent restart/recovery due to the fact that store/day records that are processed will be updated to an audit_status of A (audited), H (HQ errors pending), or S (store errors pending) and will not be fetched by the driving cursor when the program restarts.

Design Assumptions

N/A

sa_rules_eval_job (Evaluate Transactions and Totals based on Client Defined Rules)

Module Name	sa_rules_eval_job
Description	Evaluate Transactions and Totals based on Client Defined Rules
Functional Area	Oracle Retail Sales Audit
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Integration Catalog ID	NA

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (SA_RULES_EVAL_THREAD) will filter eligible records from sales audit store day (sa_store_day) table for all stores wherein auditing status is "T"otaled. Evaluating rules is integral to the sales auditing process. Rules make the comparisons between data from various sources. These comparisons find data errors that could be the result of either honest mistakes or fraud. Finding these mistakes during the sales auditing process prevents these errors from being passed on to merchandising and data warehouse systems.

Rules in ReSA are dynamic. Aside from basic data validations rules are not predefined in the system. Retailers have the ability to define through the online Rule Definition Wizard. Errors uncovered by these rules are available for review on-line during the interactive audit process. After users modify existing rules or create new ones, they become part of the rules the next time that this program runs. These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table B8D_SA_RULES_EVAL_STG.

The Business logic program (SA_RULES_EVAL) will process all records from the staging table. Using bulk processing, this program will process the records for auditing evaluation by calling SA_AUDIT_RULES_SQL.PROCESS_AUDIT_RULES for each store day captured. It will free up and clean the staging table afterwards. There is STOP ON NEXT feature in bulk processing (via loop) where Administrators can stop this batch with a flip of this indicator.

Scheduling Constraints

Table 27–55 Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad Hoc
Frequency	Daily - 2 hours interval - 5 time a day
Scheduling Considerations	NA
Pre-Processing	NA
Post-Processing	NA
Threading Scheme	Threaded by Store Day Sequence number

Restart/Recovery

NA

Key Tables Affected

Table 27–56 Key Tables Affected

Table	Select	Insert	Update	Delete
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
JOB_AUDIT_LOGS	No	Yes	No	No
B8D_SA_RULES_EVAL_STG	Yes	No	Yes	Yes
SA_STORE_DAY	Yes	No	Yes	No
SA_RULE_HEAD	Yes	No	No	No
SA_RULE_LOC_TRAIT	Yes	No	No	No
SA_ERROR_WKSHT	No	Yes	No	Yes
SA_ERROR_TEMP	No	Yes	No	No
SA_ERROR	No	Yes	Yes	Yes
SA_TOTAL	No	No	Yes	No

Table 27–56 (Cont.) Key Tables Affected

Table	Select	Insert	Update	Delete
SA_TRAN_HEAD	No	No	Yes	No
SA_TRAN_ITEM	No	No	Yes	No
SA_TRAN_DISC	No	No	Yes	No
SA_TRAN_TENDER	No	No	Yes	No
SA_TRAN_TAX	No	No	Yes	No

Design Assumptions

NA

sapreexp (Prevent Duplicate Export of Total Values from ReSA)

Module Name	sapreexp.pc
Description	Prevent Duplicate Export of Total Values from Sales Audit
Functional Area	Oracle Retail Sales Audit
Module Type	Admin
Module Technology	ProC
Catalog ID	RSA20
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

When you modify or revise a transaction through the Sales Audit user application, numerous totals may be affected and require re-totaling. The sales audit pre-export module is designed to compare the latest prioritized version of each total defined for export with the version that was previously sent to each system. If they are the same, an SA_EXPORTED entry is created for the total for that particular system, so that the same value will not be exported twice. By determining which totals have not changed since the last export date time (SA_EXPORTED_REV), this module will then create entries on SA_EXPORTED to prohibit any third-party application from receiving multiple export revisions.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Only two commits will be done. One to establish the store/day lock (this will be done by the package) and one at the end after a store/day or store/day/total has been completely processed.

Design Assumptions

N/A

saexprms (Export of POS transactions from Sales Audit to Merchandising)

Module Name	saexprms.pc
Description	Export of POS transactions from Sales Audit to Merchandising
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA01
Wrapper Script	rmswrap_multi_dnld_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this batch module is to fetch all sale and return transactions that do not have Merchandising errors from the Sales Audit database tables for transmission to the Merchandising system. Transaction data is rolled up to the item/store/day/price point/sales type level for SALES transaction type and item/store/day/price point/sales type/no inventory return indicator/return disposition/return warehouse level for RETURN transaction types.

If unit of work system parameter is defined as 'S', then the whole store/day is skipped if any Merchandising error is found. If this value is 'T', then only transactions with Merchandising errors are skipped.

If the transaction has a status of Deleted and it has previously been transmitted, a reversal of the transaction will be sent.

A file is generated for each store/day.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Records will be fetched, updated, and inserted in batches of pl_commit_max_ctr. Only two commits will be done, one to establish the store/day lock and another at the end, to release the lock after a store/day has been completely processed. The POSU formatted output file will be created with a temporary name and renamed just before the end of store/day commit.

In case of failure, all work done will be rolled back to the point right after the call to get_lock() and the lock will be released. Thus, the rollback segment should be large enough to hold all inserts into sa_exported for one store/day.

I/O Specification

Integration Type Download from Sales Audit

File Name "POSU_" appended with store number, business date and system date

Integration Contract IntCon000044

Table 27–57 File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Identifies the file record type.
	File Line Id	Char(10)	0000000001	Sequential file line number.
	File type definition	Char(4)	POSU	Identifies the file type
	File Create Date	Char(14)	N/A	File Create Date in YYYYMMDDHHMMSS format.
	Store	Number(10)	N/A	Store location.
	Vat include indicator	Char(1)	N/A	Determines whether or not the store values include VAT. Not required, but populated by Sales Audit.
	Vat region	Number(4)	N/A	VAT region the given location is in. Not required, but populated by Sales Audit.
	Currency code	Char(3)	N/A	Currency of the given location. Not required, but populated by Sales Audit.
	Currency retail decimals	Number(1)	N/A	Number of decimals supported by given the currency for retails. Not required, but populated by Sales Audit.
THEAD	Record descriptor	Char(5)	THEAD	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Transaction date	Char(14)	N/A	Transaction date in YYYYMMDDHHMMSS format. Corresponds to the date that the sale/return transaction was processed at the POS.
	Item Type	Char(3)	REF or ITM	Can be REF or ITM.
	Item	Char(25)	N/A	ID number of the ITM or REF.
	Dept	Number(4)	N/A	Department of item sold or returned.
	Class	Number(4)	N/A	Class of item sold or returned.

Table 27–57 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Sub Class	Number(4)	N/A	Subclass of item sold or returned.
	Pack Ind	Char(1)	N/A	Pack indicator of item sold or returned.
	Item Level	Number(1)	N/A	Item level of item sold or returned.
	Tran level	Number(1)	N/A	Transaction level of item sold or returned.
	Wastage Type	Char(6)	N/A	Wastage type of item sold or returned.
	Wastage pct	Number(12)	N/A	Waste pct (4 implied decimal places).
	Tran type	Char(1)	N/A	Transaction type code to specify whether transaction is a sale or a return.
	Drop Shipment indicator	Char(1)	N/A	Indicates whether the transaction is a drop shipment or not.
	Total sales qty	Number(12)	N/A	Total sales quantity (4 implied decimal places).
	Selling UOM	Char(4)	N/A	Selling Unit of Measure for the item.
	Sales sign	Char(1)	N/A	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.
	Total Sales Value	Number(20)	N/A	Total sales value of goods sold/returned (4 implied decimal places).
	Last Date time modified	Char(14)	N/A	Date and time of last modification in YYYYMMDDHHMMSS format.
	Catchweight indicator	Char(1)	N/A	Indicates if item is a catchweight item.
	Total weight	Number(12)	N/A	The actual weight of the item, only populated if catchweight_ind = Y.
	Sub Tran type indicator	Char(1)	N/A	Transction type for Sales Audit. Valid values are A, D, and NULL.
	Total IGTAX Value	Number(20)	N/A	This indicates total of all IGTAX amount for the item.

Table 27–57 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Sales Type	Char(1)	N/A	This column indicates whether the line item is a Regular Sale, a customer order serviced by OMS (External CO), or a customer order serviced by a store (In Store CO).
	No Inventory Return Indicator	Char(1)	N/A	This column contains an indicator that identifies a return without inventory. This is generally a non-required column, but in the case of Returns, this is required.
	Return Disposition	Char(10)	N/A	This column contains the disposition code published by Oracle Retail Warehouse Management System (RWMS0 as part of the Returns upload to OMS.
	Return Warehouse	Char(10)	N/A	This column contains the physical warehouse ID for the warehouse identifier where the item was returned.
	Customer Order No	Char(48)	N/A	This column contains the customer order number ID.
	Fulfillment Order No	Char(48)	N/A	This column contains the fulfillment order number ID.
	Fulfillment Loc Type	Char(2)	N/A	This column contains the fulfillment location type. Code for the fulfillment loc type from code_detail where code_type = 'FLTP'
	Fulfillment Loc	Number(10)	N/A	This column contains the fulfillment loc ID.
	Orig Store	Number(10)	N/A	This column contains the original store value for a Return transaction.
	POS Tran Id	Number(20)		This column contains the unique identifier for a sale transaction. This is an **Optional** field.
TTAX	Record descriptor	Char(5)	TTAX	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Tax Code	Char(6)	N/A	The Tax Code of the item.
	Tax Rate	Number(20)	N/A	The tax rate of the item (10 implied decimal places).

Table 27–57 (Cont.) File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Total Tax Amount	Number(20)	N/A	The item level tax or prorated transaction level tax of the item (4 implied decimal places).
TDETL	Record descriptor	Char(5)	TDETL	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Promo Tran Type	Char(6)	N/A	Code for the promotional type from code_detail where code_type equals PRMT.
	Promotion Number	Number(10)	N/A	Promotion number from Merchandising.
	Sales quantity	Number(12)	N/A	Sales quantity sold for this promotion type (4 implied decimal places).
	Sales value	Number(20)	N/A	Sales value for this promotion type (4 implied decimal places).
	Discount value	Number(20)	N/A	Discount value for this promotion type (4 implied decimal places).
	Promotion component	Number(10)	N/A	Links the promotion to additional pricing attributes. Contains the offer ID from Pricing.
TTAIL	Record descriptor	Char(5)	TTAIL	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Tran Record Counter	Number(6)	N/A	Number of TDETL records in this transaction set.
FTAIL	Record descriptor	Char(5)	FTAIL	Identifies the file record type.
	File Line Id	Number(10)	N/A	Sequential file line number.
	File Record counter	Number(10)	N/A	Number of records/transactions processed in the current file (only records between head and tail).

Fields expected in POSU format based on changes adopted:

	V16	V16 with Customer Order Changes	V19
Fulfillment Order No	No	Yes	Yes
Fulfillment Loc Type	No	Yes	Yes
Fulfillment Loc	No	Yes	Yes
Orig Store	No	Yes	Yes

	V16	V16 with Customer Order Changes	V19
POS Tran Id	No	No	Yes

Design Assumptions

- Tax can be sent either in TTAX or IGTAX regardless of the default_tax_type of SVAT, GTAX, or SALES. But prorated tax in TTAX will only be sent to Merchandising in an SVAT configuration since proration is based on VAT_ITEM and VAT_ITEM and is only defined for SVAT.
- POS can send either transactional level tax details in TTAX lines or item-level tax details in IGTAX lines through the RTLOG file to Sales Audit. These tax details will be passed on to Merchandising in the TTAX lines of the POSU file. Even though POS can pass multiple IGTAX/TTAX lines to Sales Audit and from Sales Audit to Merchandising, Merchandising only supports one tax code per item. If multiple taxes for an item are sent from POS to Sales Audit, they will be summed to a single tax in Merchandising sales upload process and assigned one of the applicable tax codes when writing tran_data 88.

saordinvexp (Export Inventory Reservation/Release for In Store Customer Order & Layaway Transactions from ReSA)

Module Name	saordinvexp.pc
Description	Export Inventory Reservation/Release for In Store Customer Order & Layaway Transactions from Sales Audit
Functional Area	Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA12
Wrapper Script	rmswrap_multi_dnld_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program will generate a flat file to reserve or un-reserve the inventory for items on in-store customer order or layaway transactions. Inventory will be reserved for items on customer order/layaway initiate and un-reserved for customer order/layaway cancel or complete transactions.

Customer orders can be categorized into two categories: In-Store Customer Orders and External Customer Orders. The In-Store Customer Orders are defined as orders that are serviced at the store and inventory reservation is done in Oracle Retail Store Inventory Management (SIM). While the External Customer orders are serviced by an external order management system, no inventory reservation will be made at the store in SIM.

This batch should only process records where the sales type is not equal to External Customer Sales, as it handles only the in-store type orders.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination.

Records are fetched, updated, and inserted in batches of pl_commit_max_ctr. Only two commits are done, one to establish the store/day lock and another at the end, to release

the lock after a store/day is completely processed. The ORIN formatted output file is created with a temporary name and renamed just before the end of store/day commit.

In case of failure, all work done is rolled back to the point right after the call to get_ lock() and the lock is released. Thus, the rollback segment should be large enough to hold all inserts into sa_exported for one store/day.

I/O Specification

Integration Type Inventory Export from Sales Audit to Merchandising

File Name ORIN_<store>_<tran_date>_<sysdate>

Integration Contract IntCon000049

Output File Layout

Table 27–58 Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Identifies the file record type.
	File Line Id	Char(10)	0000000001	Sequential file line number.
	File type Definition	Char(4)	ORIN	Identifies the file type.
	File Create Date	Char(14)	N/A	File Create Date in YYYYMMDDHHMMSS format.
	Location	Number(10)	N/A	Store location number.
THEAD	Record descriptor	Char(5)	THEAD	Identifies the file record type.
	File Line Id	Char(10)		Sequential file line number.
	Transaction Date & Time	Char(14)	Transaction Date	Date and time of the order processed.
	Transaction Type	Char(6)	SALE	Transaction type code specifies whether the transaction is sale or return.

Table 27–58 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
TDETL	Record descriptor	Char(5)	TDETL	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Item Type	Char(3)	REF or ITM	Can be REF or ITM.
	Item	Char(25)	N/A	ID number of the ITM or REF.
	Item Status	Char(6)	LIN - Layaway Initiate LCA - Layaway Cancel LCO - Layaway Complete PVLCO - Post void of Layaway complete ORI - Pickup/delivery Initiate ORC - Pickup/delivery Cancel ORD - Pickup/delivery Complete PVORD - Post void of Pick-up/delivery complete	Type of transaction.
	Dept	Number(4)	N/A	Department of item sold or returned.
	Class	Number(4)	N/A	Class of item sold or returned
	Sub class	Number(4)	N/A	Subclass of item sold or returned.
	Pack Ind	Char(1)	N/A	Pack indicator of item sold or returned.
	Quantity Sign	Chanr(1)	P or N	Sign of the quantity.
	Quantity	Number(12)	N/A	Quantity * 10000 (4 implied decimal places), number of units for the given order (item) status.
	Selling UOM	Char(4)	N/A	UOM at which this item was sold.

Table 27–58 (Cont.) Output File Layout

Record Name	Field Name	Field Type	Default Value	Description
	Catchweight Ind	Char(1)	N/A	Indicates if the item is a catchweight item. Valid values are Y or NULL.
	Customer Order number	Char(48)	N/A	Customer Order number.
TTAIL	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type.
	File Line Identifier	Number(10)	Specified by Sales Audit	ID of current line being processed by input file.
	Transaction count	Number(6)	Specified by Sales Audit	Number of TDETL records in this transaction set.
FTAIL	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type.
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.
	File Record Counter	Number(10)		Number of records/transactions processed in the current file (only records between FHEAD and FTAIL).

Design Assumptions

N/A

saexpdw (Export from Sales Audit to Oracle Retail Analytics)

Module Name	saexpdw.pc
Description	Export from Sales Audit to Oracle Retail Analytics
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA02
Wrapper Script	batch_resa2dw.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this batch module is to fetch all sales and return transactions that do not have Retail Analytics errors from the Sales Audit database tables for transmission to the Oracle Retail Analytics application. The data will be sent at the store day level. If the transaction has a status of Deleted, and if it has been previously Transmitted, a reversal of the transaction will be sent.

Note: This batch program can be run in two modes - trickle mode and batch mode. If 'Y' is passed as a parameter while running the batch program, then the batch runs in trickle mode. If 'N' or no parameter is passed, it runs in normal batch mode.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Records will be fetched, updated, and inserted based on the commit_max_ctr. Only two commits will be done: one to establish the store/day lock and another at the end, to release the lock after a store/day has been completely processed. The RDWT, RDWF, RDWS, and RDWC formatted output files will be created with temporary names and renamed just before the end of store/day commit.

In case of a failure, all the work done will be rolled back to the point right after the call to get_lock() and the lock is released. Thus, the rollback segment should be large enough to hold all inserts into sa_exported for one store/day.

I/O Specification

Integration Type	Download from Sales Audit
File Name	RDWT_ appended with store number, business date, and system date. RDWF_ appended with store number, business date, and system date. RDWS_ appended with store number, business date, and system date. RDWC_ appended with store number, business date, and system date.
Integration Contract	IntCon000041 (RDWT) IntCon000156 (RDWF) IntCon000157 (RDWS) IntCon000158 (RDWC)

Four output files will be created for each store_day:

- RDWT - Transaction File
- RDWF - Form of Payment (Tender) file
- RDWS - Store Totals output file
- RDWC - Cashier output File

Each output file is converted into a format for loading into Retail Analytics by the resa2dw Perl script.

Sales Audit - File Layout - Retail Analytics

- File layouts for the interface between sales audit and Retail Analytics.
- Char fields are left justified and blank filled.
- Number fields are right justified and zero filled. They can contain only numbers.
- Numeric fields are left justified and blank filled. They can contain only numbers.

RDWT File

Table 27–59 RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	File Type Definition	Char(4)	RDWT	Identifies file as Retail Analytics Transaction file.	Yes
	File Create Date	Number(14)	Create date	Date file was written by external system. Format YYYYMMDDHH24MISS	Yes
Transaction Header	File Type Record Descriptor	Char(5)	THEAD	Identifies transaction record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	Business date	Number(8)	N/A	Format YYYYMMDD (Note: This is the date the Retail Analytics will consider the transaction date.)	Yes
	Transaction Date	Number(14)	Transaction date	Date sale/return transaction was processed at the POS. Format YYYYMMDDHH24MISS (Note: the Retail Analytics only uses the HH24MI part of this date.)	Yes

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Location	Number(10)	Specified by external system	Store or warehouse identifier. This value is now being determined based on either the Account for Sale or Account for Return system option.	Yes
	Register ID	Char(5)	N/A	The register identifier.	Yes, -1 for null
	Banner ID	Char(4)	N/A	The unique identifier of the banner.	Yes, -1 for null
	Line Media ID	Char(10)	N/A	The identifier of the media for the order line. For non-merchandise items, such as Shipping & Handling, Service Lines. and gift certificates, the media code will be that of the order line with which it is associated.	Yes, -1 for null
	Selling Item ID	Char(25)	N/A	The unique identifier of a selling item.	Yes, -1 for null
	Customer Order Header ID	Char(48)	N/A	The unique identifier of a customer order.	Yes, -1 for null
	Customer Order Line ID	Char(30)	N/A	The identifier of a customer order line. For a Value Added Service, such as monogramming, this will be the line number for the item which the service was applied.	Yes, -1 for null
	Customer Order Create Date	Char(8)	N/A	The date when the customer order was created/placed.	Yes, -1 for null
	Cashier Identifier	Char(10)	N/A	The cashier number. This will be the unique employee number.	Yes, -1 for null
	Salesperson Identifier	Char(10)	N/A	The salesperson number. This will be the unique employee number.	Yes, -1 for null

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Customer ID Type	Char(6)	N/A	The type of ID number used by this customer.	Yes, -1 for null
	Customer ID Number	Char(16)	N/A	Customer ID associated with the transaction.	Yes, -1 for null
	Transaction Number	Number(10)	N/A	The unique transaction reference number generated by the POS.	Yes
	Original Register ID	Char(5)	N/A	Register ID of the original transaction.	Yes for a transactio n type of PVOID.
	Original Transaction Number	Number(10)	N/A	Transaction number of the original transaction.	Yes for a transactio n type of 'PVOID, EEXCGH and RETURN
	Transaction Header Number	Numeric(20)	N/A	Unique reference used within sales audit to represent the date/store/register /tran_no.	Yes
	Revision number	Number(3)	N/A	Number used to identify the version of the transaction being sent.	Yes
	Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.	Yes
	Transaction Type	Char(6)	N/A	Transaction type code.	Yes
	Sub Transaction Type	Char(6)	N/A	The Sub Transaction type.	Yes, -1 for null
	Retail Type	Char(1)	R (Regular), P (Promo), or C (Clearance)	N/A	Yes
	Item_Seq_No	Number(4)	N/A	The order in which items were entered during the transaction.	No

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Employee Number (Cashier)	Char(10)	N/A	Employee identification number. This will only be populated if the sub transaction type is EMP.	Yes, -1 for null
	Receipt Indicator	Char(1)	N/A	Flag that identifies returns that have been processed without a receipt. This field will only be populated if the transaction type is RETURN.	No
	Reason Code	Char(6)	N/A	A reason is required with a Paid In/Out transaction type, and optional with a return transaction.	Yes, -1 for null
	Vendor number	Numeric(10)	N/A	This will only get populated when the paid in code is Expense Vendor.	No
	Item Type	Char(6)	item type identifier	Type of item sold: ITEM, REF, GCN (gift certificate number), or NMITEM.	No
	Item	Char(25)	N/A	ID number of the item or gift certificate.	No. Required if Item Type is not null.
	Ref Item	Char(25)	N/A	Sub-transaction level item.	No. Also, this field can never be populated without a transaction level item in the item field.
	Taxable Indicator	Char(1)	N/A	Taxable/non-taxable status indicator.	No
	Entry/mode	Char(6)	N/A	Indicator that identifies whether the item was scanned or manually entered.	No

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Department	Number(4)	N/A	Department of item sold or returned. Need to validate if using Sales Audit.	No
	Class	Number(4)	N/A	Class of item sold or returned. Need to validate if using Sales Audit.	No
	Subclass	Number(4)	N/A	Subclass of item sold or returned. Need to validate if using Sales Audit.	No
	Total Sales Quantity	Number(12)	N/A	Number of units sold at a particular location, with 4 implied decimal places.	No
	Total Transaction Value	Number(20)	N/A	Sales value, net sales value of goods sold/returned, with 4 implied decimal places.	No
	Override Reason	Char(6)	N/A	This column will be populated when an item's price has been overridden at the POS to define why it was overridden. This will also always be sent if the transaction originated in RCOM.	Yes, -1 for null
	Return Reason	Char(6)	N/A	The reason an item was returned.	Yes, -1 for null
	Total original sign	Char(1)	'P'- positive 'N' - negative	N/A	No
	Total Original Sales Value	Number(20)	N/A	This column will be populated when the item's price was overridden at the POS and the item's original unit retail is known. This will always be written when the transaction originated in RCOM. This has 4 implied decimals.	No

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Weather	Char(6)		For transaction types of 'COND', this field will store the type of weather for the store-day.	No
	Temperature	Char(6)		For transaction types of 'COND', this field will store the type of temperature for the store-day.	No
	Traffic	Char(6)		For transaction types of 'COND', this field will store the type of traffic for the store-day.	No
	Construction	Char(6)		For transaction types of 'COND', this field will store info regarding any construction on that store-day.	No
	Drop Shipment Indicator	Char(1)	Y or N	Indicates whether the item is involved in a drop shipment.	No
	Item Status	Char(6)	N/A	The status of the item, required for voided or exchanged items. Valid values are found in the code_detail table under code_type SASI.	Y, -1 for null
	Tran Process Sys	Char(3)	N/A	This column holds the name of the system that processed the transaction. This will be used for filtering duplicate transactions coming from the different systems for export to downstream systems. Expected values are POS - Point of Sale, OMS - Order Management System and, SIM - Store Inventory Management.	Y, -1 for null

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Return Wh	Number(10)	N/A	This column contains the physical warehouse ID for the warehouse identifier where the item was returned.	N, -1 for null
	Fulfill Order No	Char(48)	N/A	This column holds the number from OMS related to the fulfillment details. One or more fulfillment orders could relate back to a single customer order in OMS. This column is required if the order is a cross channel order (that is, Sales Type equals E) and the item status is ORD.	N, -1 for null
	No Inventory Return Ind	Char(1)	N/A	This column contains an indicator that identifies a return without inventory. This is generally a non-required column, but in the case of returns, this is required.	No
	Sales Type	Char(1)	N/A	This column indicates whether the line item is a Regular Sale, a customer order serviced by OMS (External CO), or a customer order serviced by a store (In Store CO).	Yes
	Return Disposition	Char(10)	N/A	This column will contain the disposition code published by RWMS as part of the Returns upload to OMS.	N, -1 for null
	Original Store	Char(10)		This column contains the store ID for the original store.	N

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Original Transaction Number	Number(10)		Original transaction number for the returned item.	N
Transaction Detail	File Type Record Descriptor	Char(5)	TDETL	Identifies transaction record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	Discount Type	Char(6)	N/A	Code for discount type from code_detail, code_type equals SADT.	No
	Promotional Transaction Type	Char(6)	N/A	Code for promotional type from code_detail, code_type equals PRMT.	Yes
	Promotion Number	Numeric(10)	Promotion number	Promotion number from Merchandising.	No
	Promotion Component Number	Numeric(10)	Offer ID from Pricing	N/A	Required if it is a promotio nal sale.
	Coupon Number	Char(40)	N/A	N/A	Yes, if Discount Type is SCOUP.
	Coupon Reference Number	Char(16)	N/A	N/A	No
	Sales Quantity	Number(12)	N/A	Number of units sold in this promotion type, with 4 implied decimal places.	No
	Transaction Sign	Char(1)	P- positive N - negative	N/A	Yes
	Transaction Value	Number(20)	N/A	Value of units sold in this promotion type, with 4 implied decimal places.	Yes
	Discount Value	Number(20)	N/A	Value of discount given in this promotion type, with 4 implied decimal places.	Yes

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	Reference Number 1	Char(30)			No
	Reference Number 2	Char(30)			No
	Reference Number 3	Char(30)			No
	Reference Number 4	Char(30)			No
	Reference Number 5	Char(30)			No
	Reference Number 6	Char(30)			No
	Reference Number 7	Char(30)			No
	Reference Number 8	Char(30)			No
	Reference Number 13	Char(30)			No
	Reference Number 14	Char(30)			No
	Reference Number 15	Char(30)			No
	Reference Number 16	Char(30)			No
	Reference Number 25	Char(30)			No
	Reference Number 26	Char(30)			No
	Reference Number 27	Char(30)			No
Transaction Trailer	File Type Record Descriptor	Char(5)	TTAIL	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of current line being processed by input file.	Yes
	Transaction Count	Number(6)	Specified by external system	Number of TDETL records in this transaction set.	Yes
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes

Table 27–59 (Cont.) RDWT File Layout

Record Name	Field Name	Field Type	Default Value	Description	Required
	File Record Counter	Number(10)	N/A	Number of records/transactions processed in the current file (only records between head and tail).	Yes

Transaction Item Information Produced by saexpdw.pc after Translation by resa2dw

Table 27–60 File Layout

Field Name	Field Type	Default Value	Description	Required
Business date	Number(8)	N/A	Format YYYYMMDD.	Yes
Transaction Date	Number(14)	Transaction Date	Date sale/return transaction was processed at the POS. Format YYYYMMDDHH24MISS	Yes
Location	Number(10)	Specified by external system	Store or warehouse identifier. This value is now being determined based on either the Account for Sale or Account for Return system option.	Yes
Register ID	Char(5)	N/A	The register identifier.	Yes, -1 for null
Banner ID	Char(4)	N/A	The unique identifier of the banner.	Yes, -1 for null
Line Media ID	Char(10)	N/A	The identifier of the order line media. For non-merchandise items, such as Shipping & Handling, Service Lines, and gift certificates, the media code will be that of the order line with which is it is associated.	Yes, -1 for null
Selling Item ID	Char(25)	N/A	The unique identifier of a selling item.	Yes, -1 for null
Customer Order Header ID	Char(48)	N/A	The unique identifier of a customer order.	Yes, -1 for null
Customer Order Line ID	Char(30)	N/A	The identifier of a customer order line. For a Value Added Service, such as monogramming, this will be the line number for the item, which the service was applied.	Yes, -1 for null
Customer Order Create Date	Number(8)	N/A	The customer order creation date.	Yes, transaction date for null

Table 27–60 (Cont.) File Layout

Field Name	Field Type	Default Value	Description	Required
Cashier Identifier	Char(10)	N/A	The cashier number. This will be the unique employee number.	Yes, -1 for null
Salesperson Identifier	Char(10)	N/A	The salesperson number. This will be the unique employee number.	Yes, -1 for null
Customer ID Type	Char(6)	N/A	The type of ID number used by this customer.	Yes, -1 for null
Customer ID Number	Char(16)	N/A	Customer ID associated with the transaction.	Yes, -1 for null
Transaction Number	Number(10)	N/A	The unique transaction reference number generated by the POS.	Yes
Original Register ID	Char(5)	N/A	Register ID of the original transaction.	Yes for a transaction type of 'PVOID'.
Original Transaction Number	Number(10)	N/A	Transaction number of the original transaction.	Yes for a transaction type of 'PVOID'.
Transaction Header Number	Numeric(20)	N/A	Unique reference used within sales audit to represent the date/store/register/tran_no.	Yes
Revision number	Number(3)	N/A	Number used to identify the version of the transaction being sent.	Yes
Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.	Yes
Transaction Type	Char(6)	N/A	Transaction type code.	Yes
Sub Transaction Type	Char(6)	N/A	The Sub Transaction type.	Yes, -1 for null
Retail Type	Char(1)	R (Regular), P (Promo), or C (Clearance)	N/A	Yes
Item_Seq_No	Number(4)	N/A	The order in which items were entered during the transaction.	No
Employee Number (Cashier)	Char(10)	N/A	Employee identification number. This will only be populated if the sub transaction type is EMP.	Yes, -1 for null

Table 27–60 (Cont.) File Layout

Field Name	Field Type	Default Value	Description	Required
Receipt Indicator	Char(1)	N/A	Flag that identifies returns that have been processed without a receipt. This field will only be populated if the transaction type is RETURN.	No
Reason Code	Char(6)	N/A	A reason is required with a Paid In/Out transaction type, and optional with a return transaction.	Yes, -1 for null
Vendor number	Numeric(10)	N/A	This will only get populated when the paid in code is Expense Vendor	No
Item Type	Char(6)	Item type identifier	Type of item sold, ITEM, REF, GCN (gift certificate number), or IMITEM.	No
Item	Char(25)	N/A	ID number of the item or gift certificate.	No. Required if Item Type is not null.
Ref Item	Char(25)	N/A	Sub-transaction level item.	No. Also, this field can never be populated without a transaction level item in the item field.
Taxable Indicator	Char(1)	N/A	Taxable/non-taxable status indicator.	No
Entry/mode	Char(6)	N/A	Indicator that identifies whether the item was scanned or manually entered.	No
Department	Number(4)	N/A	Department of item sold or returned. Need to validate if using Sales Audit.	No
Class	Number(4)	N/A	Class of item sold or returned. Need to validate if using Sales Audit.	No
Subclass	Number(4)	N/A	Subclass of item sold or returned. Need to validate if using Sales Audit.	No
Total Sales Quantity	Number(12)	N/A	Number of units sold at a particular location, with 4 implied decimal places.	No
Total Transaction Value	Number(20)	N/A	Sales value, net sales value of goods sold/returned, with 4 implied decimal places.	No

Table 27–60 (Cont.) File Layout

Field Name	Field Type	Default Value	Description	Required
Override Reason	Char(6)	N/A	This column will be populated when an item price has been overridden at the POS to define why it was overridden. This will always be sent if the transaction originated in RCOM.	Yes, -1 for null
Return Reason	Char(6)	N/A	The reason an item was returned.	Yes, -1 for null
Total original sign	Char(1)	P- positive N - negative	N/A	No
Total Original Sales Value	Number(20)	N/A	This column will be populated when the item's price was overridden at the POS and the item's original unit retail is known. This will always be sent if the transaction originated in RCOM. This has 4 implied decimals.	No
	Weather	Char(6)	For transaction types of 'COND', this field will store the type of weather for the store-day.	No
	Temperature	Char(6)	For transaction types of 'COND', this field will store the type of temperature for the store-day.	No
	Traffic	Char(6)	For transaction types of 'COND', this field will store the type of traffic for the store-day.	No
	Construction	Char(6)	For transaction types of 'COND', this field will store info regarding any construction on that store-day.	No
Drop Shipment Indicator	Char(1)	Y or N	Indicates whether the item is involved in a drop shipment.	No
Item Status	Char(6)	N/A	The status of the item, required for voided or exchanged items. Valid values are found in the code_detail table under code_type SASI.	Y, -1 for null

Table 27–60 (Cont.) File Layout

Field Name	Field Type	Default Value	Description	Required
Tran Process Sys	Char(3)	N/A	This column holds the name of the system that processed the transaction. This will be used for filtering duplicate transactions coming from the different systems for export to downstream systems. Expected values are POS - Point of Sale, OMS - Order Management System and, SIM - Store Inventory Management.	Y, -1 for null
Return Wh	Number(10)	N/A	This column contains the physical warehouse ID for the warehouse identifier where the item was returned.	N, -1 for null
Fulfill Order No	Char(48)	N/A	This column holds the number from OMS related to the fulfillment details. One or more fulfillment orders could relate back to a single customer order in OMS. This column is required if the order is a cross channel order (that is, Sales Type equals E) and the item status is ORD.	N, -1 for null
No Inventory Return Ind	Char(1)	N/A	This column contains an indicator that identifies a return without inventory. This is generally a non-required column, but in case of returns, this is required.	N
Sales Type	Char(1)	N/A	This column indicates whether the line item is a Regular Sale, a customer order serviced by OMS (External CO), or a customer order serviced by a store (In Store CO).	Y
Return Disposition	Char(10)	N/A	This column will contain the disposition code published by RWMS as part of the Returns upload to OMS.	N, -1 for null
Original Store	Char(10)	N/A	This column contains the store ID for the original store.	No
Original Transaction Number	Number(10)	N/A	Original transaction number for the returned item.	No
Discount Type	Char(6)	N/A	Code for discount type from code_detail, code_type equals SADT.	No

Table 27–60 (Cont.) File Layout

Field Name	Field Type	Default Value	Description	Required
Promotional Transaction Type	Char(6)	N/A	Code for promotional type from code_detail, code_type equals PRMT.	Yes
Promotion Number	Numeric(10)	Promotion number	Promotion number from Merchandising.	No
Promotion Component Number	Numeric(10)	Offer ID from Pricing	N/A	Required if it is a promotional sale.
Coupon Number	Char(40)	N/A	N/A	Yes if Discount Type is SCOUP.
Coupon Reference Number	Char(16)	N/A	N/A	No
Sales Quantity	Number(12)	N/A	Number of units sold in this promotion type, with 4 implied decimal places.	No
Transaction Sign	Char(1)	P - positive N - negative	N/A	Yes
Transaction Value	Number(20)	N/A	Value of units sold in this promotion type, with 4 implied decimal places.	Yes
Discount Value	Number(20)	N/A	Value of discount given in this promotion type, with 4 implied decimal places.	Yes
Reference Number 1	Char(30)			No
Reference Number 2	Char(30)			No
Reference Number 3	Char(30)			No
Reference Number 4	Char(30)			No
Reference Number 5	Char(30)			No
Reference Number 6	Char(30)			No
Reference Number 7	Char(30)			No
Reference Number 8	Char(30)			No
Reference Number 13	Char(30)			No
Reference Number 14	Char(30)			No

Table 27–60 (Cont.) File Layout

Field Name	Field Type	Default Value	Description	Required
Reference Number 15	Char(30)			No
Reference Number 16	Char(30)			No
Reference Number 25	Char(30)			No
Reference Number 26	Char(30)			No
Reference Number 27	Char(30)			No

RDWF File

Table 27–61 RDWF File

Record Name	Field Name	Field Type	Default Value	Description	Required
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	File Type Definition	Char(4)	RDWF	Identifies the file as a Retail Analytics Form of Payment (Tender) file.	Yes
	File Create Date	Numeric(14)	Create date	Date the file was written by external system. Format YYYYMMDDHH24MISS.	Yes
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	Business date	Numeric(8)	N/A	Format YYYYMMDD	Yes
	Transaction Date	Numeric(14)	Transaction date	Date sale/return transaction was processed at the POS. Format YYYYMMDDHH24MISS	Yes
	Location	Number(10)	Specified by external system	Store or warehouse identifier.	Yes

Table 27–61 (Cont.) RDWF File

Record Name	Field Name	Field Type	Default Value	Description	Required
	Cashier Identifier	Char(10)	N/A	The cashier number. This will be the unique employee number.	Yes, -1 for null
	Register Identifier	Char(5)	N/A	N/A	Yes, -1 for null
	Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.	Yes
	Transaction Sequence Number	Numeric(20)	N/A	Unique reference used within sales audit to represent the date/store/register /transaction number.	Yes
	Revision number	Number(3)	N/A	Number used to identify the version of the transaction being sent.	Yes
	Transaction Type	Char(6)	N/A	Transaction type code.	Yes
	Tender type group	Char(6)	N/A	N/A	Yes
	Tender type id	Numeric(6)	N/A	Tender type code.	Yes
	Tender amount	Number(20)	N/A	Tender amount.	Yes
	Credit Card Entry Mode	Char(6)	N/A	Contains the method in which the transaction was entered at the POS. Possible entry modes could include: Terminal Used, Magnetic Strip Track One Read, Magnetic Strip Two Read, Magnetic Strip One Transmitted, or Magnetic Strip Two Transmitted. The code type for this field is CCEM.	No
	Voucher Number	Char(25)	N/A	N/A	No
	Voucher Age	Numeric(5)	N/A	Age of the gift certificate. Redeemed date minus sold date.	Yes if Tender Type Group is VOUCH.

Table 27–61 (Cont.) RDWF File

Record Name	Field Name	Field Type	Default Value	Description	Required
	Escheat Date	Numeric(5)	N/A	Date on which this gift certificate escheats. Format is YYYYMMDD.	Yes if voucher can escheat.
	Coupon Number	Char(40)	N/A	N/A	Yes if Tender Type Group is COUPON.
	Coupon Reference Number	Char(16)	N/A	N/A	No. Only if Tender Type Group is COUPON.
	Transaction Status	Char(1)		Determines if the transaction is Present ('P') or Voided/Deleted ('R' - Reverse)	No
	Reference Number 9	Char(30)			No
	Reference Number 10	Char(30)			No
	Reference Number 11	Char(30)			No
	Reference Number 12	Char(30)			No
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	File Record Counter	Number(10)	N/A	Number of records/transactio n processed in the current file (only records between head and tail).	Yes

Retail Analytics Form of Payment File after Translation by resa2dw

Table 27–62 Form of Payment File

Field Name	Field Type	Default Value	Description	Required
Business date	Numeric(8)	N/A	Format YYYYMMDD	Yes
Transaction Date	Numeric(14)	Transaction date	Date the sale/return transaction was processed at the POS. Format YYYYMMDDHH24MISS	Yes

Table 27–62 (Cont.) Form of Payment File

Field Name	Field Type	Default Value	Description	Required
Location	Number(10)	Specified by external system	Store or warehouse identifier.	Yes
Cashier Identifier	Char(10)	N/A	The cashier number. This will be the unique employee number.	Yes, -1 for null
Register Identifier	Char(5)	N/A	N/A	Yes, -1 for null
Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.	Yes
Transaction Sequence Number	Numeric(20)	N/A	Unique reference used within sales audit to represent the date/store/register/transaction number.	Yes
Revision number	Number(3)	N/A	Number used to identify the version of the transaction being sent.	Yes
Transaction Type	Char(6)	N/A	Transaction type code.	Yes
Tender type group	Char(6)	N/A	N/A	Yes
Tender type id	Numeric(6)	N/A	Tender type code.	Yes
Tender amount	Number(20)	N/A	Tender amount.	Yes
Credit Card Entry Mode	Char(6)	N/A	Contains the method in which the transaction was entered at the POS. Possible entry modes could include: Terminal Used, Magnetic Strip Track One Read, Magnetic Strip Two Read, Magnetic Strip One Transmitted, or Magnetic Strip Two Transmitted. The code type for this field is CCEM.	No
Voucher Number	Char(25)	N/A	N/A	No
Voucher Age	Numeric(5)	N/A	Age of the gift certificate. Redeemed date minus sold date.	Yes if Tender Type Group is VOUCH.
Escheat Date	Numeric(8)	N/A	Date on which this gift certificate escheats. Format is YYYYMMDD.	Yes if voucher can escheat.
Coupon Number	Char(40)	N/A	N/A	Yes if Tender Type Group is COUPON.

Table 27–62 (Cont.) Form of Payment File

Field Name	Field Type	Default Value	Description	Required
Coupon Reference Number	Char(16)	N/A	N/A	No. Only if Tender Type Group is COUPON.
Transaction Status	Char(1)	N/A	Determines if the transaction is Present ('P') or Voided/Deleted ('R' - Reverse)	No
Reference Number 9	Char(30)			No
Reference Number 10	Char(30)			No
Reference Number 11	Char(30)			No
Reference Number 12	Char(30)			No

RDWS File

Table 27–63 RDWS File

Record Name	Field Name	Field Type	Default Value	Description	Required
File Header	File Type Record Descripto	Char(5)	FHEAD	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	File Type Definition	Char(4)	RDWS	Identifies file as a Retail Analytics Store Totals file.	Yes
	File Create Date	Char(4)	Create date	Date file was written by the external system. Format YYYYMMDDHH24MISS	Yes

Table 27–63 (Cont.) RDWS File

Record Name	Field Name	Field Type	Default Value	Description	Required
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies the transaction record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	Business date	Number(8)	N/A	Format YYYYMMDD	Yes
	Location	Number(10)	Specified by external system	Store or warehouse identifier.	Yes
	Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.	Yes
	Total ID	Char(10)	N/A	Category identifier used to determine the type of total.	Yes
	Reference Number 1	Char(30)	N/A	N/A	No
	Reference Number 2	Char(30)	N/A	N/A	No
	Reference Number 3	Char(30)	N/A	N/A	No
	Total Sign	Char(1)	P - positive N - negative	N/A	Yes
	Total Amount	Number(20)	N/A	Total over/short amount, with 4 implied decimal places.	Yes
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies the file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	File Record Counter	Number(10)	N/A	Number of records/transactions processed in the current file (only records between head and tail).	Yes

Store Totals Information after Translation by resa2dw

Table 27–64 Store Totals Information

Field Name	Field Type	Default Value	Description	Required
Business date	Number(8)	N/A	Format YYYYMMDD	Yes
Location	Number(10)	Specified by external system	Store or warehouse identifier.	Yes
Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.	Yes
Total ID	Char(10)	N/A	Category identifier used to determine the type of total.	Yes
Reference Number 1	Char(30)	N/A	N/A	No
Reference Number 2	Char(30)	N/A	N/A	No
Reference Number 3	Char(30)	N/A	N/A	No
Total Sign	Char(1)	P - positive N - negative	N/A	Yes
Total Amount	Number(20)	N/A	Total over/short amount, with 4 implied decimal places.	Yes

RDWC File

Table 27–65 RDWC File

Record Name	Field Name	Field Type	Default Value	Description	Required
File Header	File Type Record Descriptor	Char(5)	FHEAD	Identifies file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	File Type Definition	Char(4)	RDWC	Identifies the file as a Retail Analytics Cashier/Register Totals file.	Yes
	File Create Date	Numeric(14)	Create date	Date the file was written by the external system. Format YYYYMMDDHH24MISS	Yes

Table 27–65 (Cont.) RDWC File

Record Name	Field Name	Field Type	Default Value	Description	Required
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies the transaction record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	Business date	Number(8)	N/A	Format YYYYMMDD	Yes
	Location	Number(10)	Specified by external system	Store or warehouse identifie	Yes
	Cashier Identifier	Char(10)	N/A	The cashier number.	If Cashier_id is NULL, then Register_id has value. If Cashier_id has value, then Register_id is NULL. Yes, -1 for null
	Register ID	Char(5)	N/A	The register identifier.	If Cashier_id is NULL, then Register_id has value. If Cashier_id has value, then Register_id is NULL. Yes, -1 for null
	Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative	Yes
	Total ID	Char(10)	N/A	Category identifier used to determine the type of total.	Yes
	Reference Number 1	Char(30)	N/A	N/A	No
	Reference Number 2	Char(30)	N/A	N/A	No
	Reference Number 3	Char(30)	N/A	N/A	No

Table 27–65 (Cont.) RDWC File

Record Name	Field Name	Field Type	Default Value	Description	Required
	Total Sign	Char(1)	P - positive N - negative	N/A	Yes
	Total Amount	Number(20)		Total over/short amount, with 4 implied decimal places.	Yes
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies the file record type.	N/A
	File Line Identifier	Number(10)	Specified by external system	ID of the current line being processed by input file.	Yes
	File Record Counter	Number(10)	N/A	Number of records/transactions processed in the current file (only records between head and tail).	Yes

Cashier/ Register Totals Information after Translation by resa2dw

Table 27–66 Cashier/Register Totals Information

Field Name	Field Type	Default Value	Description	Required
Business date	Number(8)	N/A	Format YYYYMMDD	Yes
Location	Number(10)	Specified by external system	Store or warehouse identifier	Yes
Cashier Identifier	Char(10)	N/A	The cashier number	If Cashier_id is NULL, then Register_id has value. If Cashier_id has value, then Register_id is NULL. Yes, -1 for null

Table 27–66 (Cont.) Cashier/Register Totals Information

Field Name	Field Type	Default Value	Description	Required
Register ID	Char(5)	N/A	The register identifier.	If Cashier_id is NULL, then Register_id has value. If Cashier_id has value, then Register_id is NULL. Yes, -1 for null
Sales Sign	Char(1)	P - positive N - negative	Determines if the Total Sales Quantity and Total Sales Value are positive or negative.	Yes
Total ID	Char(10)	N/A	Category identifier used to determine the type of total.	Yes
Reference Number 1	Char(30)	N/A	N/A	No
Reference Number 2	Char(30)	N/A	N/A	No
Reference Number 3	Char(30)	N/A	N/A	No
Total Sign	Char(1)	P - positive N - negative	N/A	Yes
Total Amount	Number(20)	N/A	Total over/short amount, with 4 implied decimal places.	Yes

Design Assumptions

N/A

saexpsim (Export of Revised Sale/Return Transactions from ReSA to SIM)

Module Name	Saexpsim.pc
Description	Export of Revised Sale/Return Transactions from Sales Audit to SIM
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA14

Wrapper Script rmswrap_multi_dnld_in.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this batch module is to fetch all revised sale and return transactions that do not have SIM errors from the Sales Audit database tables for transmission to SIM. It retrieves all quantity revision transaction data for SALES, RETURN, EEXCH, VOID, and SPLORD transaction types.

If sa_system_options.unit_of_work is S, the whole store/day is skipped if any SIM error is found. If this value is T, then only transactions with SIM errors are skipped.

The batch will only export transactions whose quantity has been revised. The batch will write these revised transactions to the output file along with a reversal of the quantity.

A file of type SIMT is generated for each store/day.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Records will be fetched, updated, and inserted in batches of pl_commit_max_ctr. Only two commits will be done, one to establish the store/day lock and another at the end, to release the lock after a store/day has been completely processed. The SIMT formatted output file will be created with a temporary name and renamed just before the end of store/day commit.

In case of failure, all work done will be rolled back to the point right after the call to get_lock() and the lock released. Thus, the rollback segment should be large enough to hold all inserts into SA_EXPORTED for one store/day.

I/O Specification

Integration Type	Download from Sales Audit
File Name	SIMT_ appended by store number, business date, and system date
Integration Contract	IntCon000045

Output File Layout

Table 27–67 Output File

Record Name	Field Name	Field Type	Default Value	Description
FHEAD	Record descriptor	Char(5)	FHEAD	Identifies the file record type.
	File Line Id	Char(10)	0000000001	Sequential file line number.
	File type definition	Char(4)	SIMT	Identifies the file type.
	Store	Number(10)	N/A	Store location.
	Business Date	Char(8)	N/A	Business Date in YYYYMMDD format.
	File Create Date	Char(14)	N/A	File Create Date in YYYYMMDDHHMMSS format.
THEAD	Record descriptor	Char(5)	THEAD	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Transaction Number	Number(10)	N/A	Transaction Identifier.
	Revision Number	Number(3)	N/A	Revision Number of the transaction.
	Transaction date	Char(14)	N/A	Transaction date in YYYYMMDDHHMMSS format. Corresponds to the date that the transaction occurred.
	Transaction Type	Char(6)	N/A	Transaction Type.
	POS Transaction Indicator	Char(1)	N/A	Indicates if the transaction was received from POS or manually created. Valid values: Y - POS N - Manual

Table 27–67 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
TDETL	Record descriptor	Char(5)	TDETL	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Item Sequence Number	Number(4)	N/A	Item sequence number.
	Item	Char(25)	N/A	Identifies the merchandise item.
	Item number type	Char(6)	N/A	Identifies the type of item number if the item type is ITEM or REF.
	Item Status	Char(6)	N/A	Status of the item within the transaction, V for item void, S for sold item, R for returned item. ORI - Order Initiate ORC - Order Cancel ORD - Order Complete LIN - Layaway Initiate LCA - Layaway Cancel LCO - Layaway Complete
	Serial Number	Char(128)	N/A	Unique ID.
	Pack Indicator	Char(1)	N/A	Pack Indicator.
	Catchweight Indicator	Char(1)	N/A	Catchweight Indicator.

Table 27–67 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
	Quantity Sign	Char(1)	N/A	Sign of the quantity.
	Quantity Value	Number(12)	N/A	Number of items, with 4 implied decimal places.
	Standard Unit of Measure	Char(4)	N/A	Standard Unit of Measure of the item.
	Selling Unit of Measure	Char(4)	N/A	Unit of Measure of the quantity value.
	Waste Type	Char(6)	N/A	Waste Type.
	Waste Percent	Number(12)	N/A	Waste Percent.
	Drop Ship Indicator	Char(1)	N/A	Indicates whether the item is part of a drop shipment.
	Actual Weight	Number(12)	N/A	Contains the weight of the item sold, with 4 implied decimal places.
	Actual Weight Sign	Char(1)	N/A	Sign of the actual weight.
	Reason Code	Char(6)	N/A	Reason entered by the cashier for some transaction types.
	Sales Value	Number(20)	N/A	Transaction value, with 4 implied decimal places
	Sales Value Sign	Char(1)	N/A	Transaction value sign.
	Unit Retail	Number(20)	N/A	Unit retail, with 4 implied decimal places.
	Sales Type	Char(1)	N/A	Indicates if the transaction is an In Store Customer Order, External Customer Order, or Regular Sale.
	Customer Order Number	Char(48)	N/A	Contains the customer order ID.
	Customer Order Type	Char(6)	N/A	Customer order type.
	Fulfillment Order Number	Char(48)	N/A	Contains the order ID of the fulfillment order.
TTAIL	Record descriptor	Char(5)	TTAIL	Identifies the file record type.
	File Line Id	Char(10)	N/A	Sequential file line number.
	Tran Record Counter	Number(6)	N/A	Number of TDETL records in this transaction set.

Table 27–67 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
FTAIL	Record descriptor	Char(5)	FTAIL	Identifies the file record type.
	File Line Id	Number(10)	N/A	Sequential file line number.
	File Record counter	Number(10)	N/A	Number of records/transactions processed in the current file (only records between head and tail).

Design Assumptions

N/A

saexpim (Export DSD and Escheatment from Sales Audit to Invoice Matching)

Module Name	saexpim.pc
Description	Export DSD and Escheatment from Sales Audit to Invoice Matching
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA04
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this program is to support interfacing invoices from Direct Store Delivery and Escheatment sales audit transactions to the Invoice Matching application. Direct Store Delivery invoices refer to products or services that are delivered to the store and paid for at the store. This program will take DSD invoices that have been staged to the transaction header table by the saimptlog.pc program and move them into the invoice header table. All DSD transactions will be assumed paid. They can be assumed received if there is a proof of delivery number listed on them. Transactions with a vendor invoice ID or a proof of delivery number should be matched to any existing invoice in the invoice header, and that invoice updated with the new information being interfaced. Invoices that do not match an existing invoice in the invoice head table will need to be inserted. Each transaction will be exported to the invoice head table only once.

The Sales Audit Transaction type used to identify invoices for Direct Store Delivery transactions will be “Paid Out”. The Paid Out transaction has a code of 'PAIDOU'. The Sales Audit sub-transaction types will be used to identify whether the invoice is an

"Expense Vendor Payout" or a "Merchandise Vendor Payout". The codes are 'EV' for Expense Vendor Payout and 'MV' for Merchandise Vendor Payout. Any Paid Out transaction with a sub transaction type of Expense Vendor will create a non-merchandise invoice and cause a record to be written to the invoice non-merchandise table. Sales Audit will store non-merchandise codes in the reason_code field on sa_tran_head. Valid values for these reason codes should correspond to the codes stored on the non_merch_code_head table.

In addition to DSD invoices, this program will also interface Escheatment totals to Invoice Matching. Escheatment is the process where an unredeemed gift certificate/voucher or credit voucher will, after a set period of time, be paid out as income to the issuing retailer, or in some states, the state receives this escheatment income. Sales Audit will be the governing system that determines who receives this income, but Invoice Matching will send the totals, with the related Partner, to an Accounts Payable system. Escheatment information will be stored on the Sales Audit SA_TOTALS table and will be used to create non-merchandise invoices in Invoice Matching. These invoices will be assumed not paid.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Records will be fetched, updated, and inserted based on the commit_max_ctr specified on the restart_control table. Only two commits will be done, one to establish the store/day lock and another at the end, to release the lock after a store/day has been completely processed.

In case of failure, all work done will be rolled back to the point right after the call to get_lock and releases the lock. Thus, the rollback segment should be large enough to hold all inserts into sa_exported for one store_day.

Integration Contract

Integration Type	Download from Sales Audit
File Name	N/A
Integration Contract	IntCon00004 INVC_HEAD table

Design Assumptions

N/A

saexpgl (Post User Defined Totals from Sales Audit to General Ledger)

Module Name	saexpgl.pc
Description	Post User Defined Totals from Sales Audit to General Ledger
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Integration Catalog ID	RSA09

Wrapper Script rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this module is to post all properly configured user-defined Sales Audit totals to a general ledger application (Oracle or PeopleSoft). Totals without errors will be posted to the appropriate accounting ledger, as defined in the Sales Audit GL cross-reference module. Depending on the unit of work system parameter, the data will be sent at either the store/day or individual total level. Newly revised totals, that have already been posted to the ledger, will have their previous revision reversed, and the new total posted to the appropriate accounts. Transactions that are from previous periods will be posted to the current period.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Records will be fetched, updated, and inserted in batches the size of commit max counter. Only one commit will be done after a store/day has been completely processed. A call to release_lock() performs a commit.

I/O Specification

Integration Type	Download from Sales Audit
File Name	N/A
Integration Contract	IntCon000019 TG_FIF_GL_DATA

Design Assumptions

N/A

ang_saplgen (Extract of POS Transactions by Store/Date from Sales Audit for Web Search)

Module Name	ang_saplgen.pc
Description	Extract of POS Transactions by Store/Date from Sales Audit for Web Search
Functional Area	Sales Audit
Module Type	Integration
Module Technology	ProC
Integration Catalog ID	RMS162
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The purpose of this batch module is to fetch all corrected sale and return transactions that do not have Merchandising errors from the Sales Audit database tables for transmission to an external web search engine. If the transaction has a status of Deleted or Post Voided and has previously been transmitted, a reversal of the transaction will be sent. A file of type POSLOG is generated for each store/day.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Records will be fetched, in batches of pl_commit_max_ctr. The POSLOG formatted output file will be created with a completion of store/day looping.

I/O Specification

Integration Type Download from Sales Audit

File Name POSLOG_<store>_<business date>_<system date>.xml

Integration Contract IntCon000018

Output File Layout

Table 27–68 Output File Layout

Field Name	Field Type	Description
BatchID	CHAR(18)	A concatenation of store number and business date for a store.
RetailStoreID	CHAR(10)	The store number for which the POSLog file has to be extracted.
WorkStationID	CHAR(5)	RegistryID for the store.
TillID	CHAR(5)	RegistryID for the store.
SequenceNumber	CHAR(10)	Point of Sale system defined transaction number associated with a transaction.
BeginDate	CHAR(8)	Starting date time of the transaction.
EndDate	CHAR(8)	End date time of the transaction.
CurrencyCode	CHAR(3)	Code of the currency used during the transaction.
VoidFlag	CHAR(5)	Indicates if the item in the transaction is voided or not. Valid values are TRUE and FALSE.
Item_Status	CHAR(40)	Status of the item is required for voided, exchanged, or returned item.
MerchandisingHierarchy	CHAR(4)	Department number to which the item belongs
Description	CHAR(250)	Item description that has been sold.

Table 27–68 (Cont.) Output File Layout

Field Name	Field Type	Description
Item	CHAR(25)	Item number.
TaxIncludedInPrice	CHAR(5)	Indicates if the item is being taxed or not. Valid values are TRUE and FALSE.
RegularSalesUnitPrice	CHAR(20)	Field holds the unit retail in the standard unit of retail for the item/location combination.
ActualSalesUnitPrice	CHAR(20)	Retail price for the item.
ExtendedAmount	CHAR(20)	Total sales for the item in the detail level.
Qty	CHAR(21)	Unit sold of the item.

Design Assumptions

N/A

saescheat (Download of Escheated Vouchers from Sales Audit for Payment)

Module Name	saescheat.pc
Description	Download of Escheated Vouchers from Sales Audit for Payment
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA05
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The laws of individual states and countries may require a retailer to return monies for aged, unclaimed gift certificates, and vouchers. This process is called escheatment. This program writes records for this data to tables that are read into Invoice Matching by the program saexpim.pc. The data can then be sent as invoices approved for payment to a financial application.

The saescheat batch program will set the status of vouchers that have met certain state's escheats rules or have expired to the proper status and produce a total for later export to Invoice Matching. The rules for escheatment are defined on the sa_escheatment_options table.

Restart/Recovery

The logical unit of work is a store/day. The program commits when the number of store/day records processed has reached the commit_max_ctr.

I/O Specification

Integration Type	Download from Sales Audit
File Name	N/A
Integration Contract	IntCon000039

Design Assumptions

N/A

saescheat_nextesn (Generate Next Sequence for Escheatment Processing)

Module Name	saescheat_nextesn.pc
Description	Generate Next Sequence for Escheatment Processing
Functional Area	Oracle Retail Sales Audit
Module Type	Admin
Module Technology	ProC
Catalog ID	RSA25
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This batch program gets the next free sequence for use in the saescheat.pc process. This routine goes and gets a block of numbers when starting, and parcels them out as needed. Once they are all used up, it gets another block and returns a pointer to the string containing the next available number or NULL if an error occurs. This process is executed as part of the saexcheat.pc processing.

Restart/Recovery

NA

Design Assumptions

N/A

saexpach (Download from Sales Audit to Account Clearing House (ACH) System)

Module Name	saexpash.pc
Description	Download from Sales Audit to Account Clearing House (ACH) System
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA03
Wrapper Script	rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This module will post store/day deposit totals to the SA_STORE_ACH table and bank deposit totals for a given day in a file formatted for export to an ACH (Account Clearing House). The ACH export deviations from the typical Sales Audit export in that store/days must be exported even though errors may have occurred for a given day or store (depending on the unit of work defined), and also, the store/day does not need to be closed for the export to occur. The nature of the ACH process is such that as much money as possible must be sent as soon as possible to the consolidating bank. Any adjustments to the amount sent can be made using the sabnkach screen in the online system.

Deposits for store/days that have not been Fully (F) loaded will not be transferred to the consolidating bank. After they are fully loaded, their deposits will be picked up by the next run of this program.

Restart/Recovery

This module is in two distinct parts, with two different logical units of work. Thus, restart/recovery has to be implemented so that the first part does not get reprocessed in case the program is being restarted. Details on the implementation follow.

The first driving cursor in this module retrieves a store/day to generate ACH totals. Once the first cursor is complete, the second retrieves bank locations by account numbers.

The first Logical Unit of Work (LUW) is defined as a unique store/day combination. Records will be fetched, using the first driving cursor, in batches of commit_max_ctr, but processed and committed one store/day at a time.

The first driving cursor will fetch all store/days that have been Fully Loaded (F), whose audit status is Audited (A), HQ Errors Pending (H), or Store Errors Pending (S) and that are ready to be exported to ACH. Before processing starts, a write lock is obtained using get_lock (). This driving cursor only fetches store/days with a sa_export_log.status of SAES_R. After a store/day is processed, sa_export_log.status is set to SAES_P so that this store/day will not be selected again if the program is restarted.

The commit is performed using retek_force_commit after each store/day has been processed and sa_export_log updated, so as to release the lock.

In case a store/day could not be processed due to locking, the store/day information is placed on a list (called locked store/day list) and the next store/day is processed. This list is kept in memory and is available only during processing. If the store for a store/day obtained from the first driving cursor, is on the locked store/day list, then this store/day cannot be processed. This is the case because there is a data dependency such that data from a particular store/day is dependent on data for the same store but at an earlier date. Thus, if a store/day cannot be processed, then subsequent store/days for the same store cannot be processed either. After the driving cursor returns no more data, the program attempts to process each store/day on the list two more times. If the store/day is still locked, then it is skipped entirely and a message is printed to the error log.

The second LUW is a bank account number. Again, records will be fetched in batches of commit_max_ctr. The second driving cursor cannot retrieve information by the LUW because it is possible for the store's currency to be different from the local bank's currency. In that case, a currency conversion is needed.

For each store/day, the query should retrieve the required ACH transfer. The latter is determined by adding the estimated deposit for the next day, the adjustment to the estimate for the current day, and any manual adjustment to the estimate.

Since a store can be associated with different accounts at different banks, only accounts that are consolidated should be retrieved. Since it is possible for the local bank to be in a different country than the consolidating bank, the currency of the partner should also be fetched.

Since processing is dependent on the type of account at the RDFI, the account type should be fetched by this cursor.

Due to differences in transaction processing in cases when the bank is outside the United States, the partner's country should also be fetched. The results of the query should be sorted by partner country. The results of the query should also be ordered by accounts.

Security Considerations

The fact that this program automates the transfer of funds on behalf of the user makes it a likely target for electronic theft. It must be made clear that the responsibility of electronic protection lies with the users themselves.

Following are some tips and recommendation to users:

- A specific user should be used to run the program. This user would be the only one (or one of a few) who has access to this program.
- The umask for this user should be set up so as to prevent other users from reading/writing its files. This would ensure that when the output file is created, it would not be accessible to other users.
- The appropriate permissions should be set up on the directory, which holds the ACH files. The most restrictive decision would be to not allow any other user to view the contents of the directory.
- A secure means of communication should be implemented for transferring the file from where it has been created to the ACH network. This may be done through encryption, or by copying the file to a disk and trusting the courier to deliver the files intact.

- The ACH network needs to be secure.

I/O Specification

Integration Type Download from Sales Audit

File Name ACH_ appended with the consolidating routing number, consolidating account number, and current system date.

Integration Contract IntCon000040

Output File

Table 27–69 Output File

Record Name	Field Name	Field Type	Default Value	Description
ACH File Header	Section No.	Number(3)	101	Constant number.
	Console Route No	Number(10)	N/A	The routing number of the consolidating bank.
	Sender ID	Char(10)	N/A	ID used by the Originator to identify itself.
	Current Date	Char(6)	N/A	Vdate in YYMMDD format.
	Day Time	Char(4)	N/A	Time of file creation in HH24MM format.
	File Header No.	Number(7)	0094101	Constant number.
	Console Bank Name	Char(23)	N/A	Name of the Originating Financial Depository Institution.
	Company Name	Char(23)	N/A	The name of the company name.
	Ref Code	Char (8)	N/A	Reference code.

Table 27–69 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
ACH CCD Batch Header	Section No.	Number(4)	5225	Constant number.
	Company Name	Char(16)	N/A	The name of the company.
	Comp Disc Data	Char(20)	NULL	Any kind of data specific to the company.
	Comp Id	Char(10)	N/A	Alphanumeric code to identify the company.
	CCD Header Id	Char(3)	CCD	Constant value.
	Comp Entry Desc	Char(10)	CONSOL	A short description from the Originator about the purpose of the entry.
	Tomorrow	Char(6)	N/A	Vdate+1 in YYMMDD format.
	Tomorrow	Char(6)	N/A	Vdate+1 in YYMMDD format.
	Settle Date	Char(3)	NULL	This is inserted by receiving the ACH Operator.
	Reserved	Number(1)	1	Constant number.
	Odfi Id	Number(8)		8-digit routing number of the ODFI.
	Batch No	Number(7)		Batch number.

Table 27–69 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
ACH CBR Batch Header	Section No.	Number(4)	5225	Constant number.
	Company Name	Char(16)	N/A	The name of the company.
	Reserved	Char(3)	FV1	Constant value.
	Exch Rate	Number(15)		Exchange rate for the specified currency.
	Reserved	Char(2)	US	Constant value.
	Comp Id	Char(10)		Alphanumeric code to identify the company
	CBR Header Id	Char(3)	CBR	Constant value.
	Comp Entry Desc	Char(10)	"CONSOL "	A short description from the Originator about the purpose of the entry.
	Partner Curr Code	Char(3)	N/A	Code identifying the currency the partner uses for business transactions.
	Reserved	Char(3)	USD	Constant value.
	Tomorrow	Char(6)	N/A	Vdate+1 in YYMMDD forma.
	Settle Date	Char(3)	NULL	This is inserted by the receiving ACH Operator.
	Reserved	Number(1)	1	Constant number.
	Odfi Id	Number(8)	N/A	8-digit routing number of the ODFI.
	Batch No	Number(7)	N/A	Batch number.

Table 27–69 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
ACH CCD Entry	Section No.	Number(1)	6	Constant number.
	Trans Code	Char(2)		Code used to identify the type of debit and credit. Value accepted are 27 and 37.
	Routing No	Number(9)		Routing number for the bank account.
	Acct No	Char(17)		Account number of the bank.
	Deposit	Number(10)		The amount involved in the transaction* 10000 (4 implied decimal places).
	Id	Char(15)	Null	Identification number. Optional field containing a number used by the Originator to insert its own number for tracing purposes.
	Store Name	Char(22)		Name of the local store.
	Disc Data	Char(2)	Null	Discretionary data. Any kind of data specific to the transaction.
	Reserved	Number(1)	0	Constant number.
	Trace No	Number(15)		Used to uniquely identify each entry within a batch. The first 8 digits contain the routing number of the ODFI and the other 7 contains a sequence number.

Table 27–69 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
ACH CBR Entry	Section No.	Number(1)	6	Constant number.
	Trans Code	Char(2)	N/A	Code used to identify the type of debit and credit. Values accepted are 27 and 37.
	Routing No	Number(9)	N/A	Routing number for the bank account.
	Acct No	Char(17)	N/A	Account number of the bank
	Deposit	Number(10)	N/A	The amount involved in the transaction* 10000 (4 implied decimal places).
	Id	Char(15)	NULL	Identification number. Optional field containing a number used by the Originator to insert its own number for tracing purposes.
	Store Name	Char(22)	N/A	Name of the local store.
	Disc Data	Char(2)	NULL	Discretionary data. Any kind of data specific to the transaction.
	Reserved	Number(1)	1	Constant number.
	Trace No	Number(15)	N/A	Used to uniquely identify each entry within a batch. The first 8 digits contain the routing number of the ODFI and the other 7 contains a sequence number.
ACH CBR Addendum	Section No.	Number(3)	701	Constant number.
	Payment Info	Char(80)	Null	Payment related information.
	Reserved	Number(4)	0001	Constant number
	Trace Seq No	Number(7)	N/A	Sequence number part of the Trace Number of the entry record to which this addendum is referring.

Table 27–69 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
ACH Batch Control	Section No.	Number(4)	8225	Constant number.
	Batch Line Count	Number(6)	N/A	The number of entries and addenda in the batch.
	Hash Count	Number(10)	N/A	Sum of the RDFI IDs in the detail records.
	Total Batch Debit	Number(12)	N/A	Contains the accumulated debit and debit for the file * 10000 (4 implied decimal places).
	Total Batch Credit	Number(12)	N/A	Contains the accumulated credit and credit for the file * 10000 (4 implied decimal places).
	Comp Id	Char(10)	N/A	An alphanumeric code identifying the company.
	Auth	Char(19)	Null	Message Authentication Code. The first 8 characters represent a code from the Data Encryption Standard (DES) algorithm. The remaining eleven characters are blanks.
	Reserved	Char(6)	Null	Reserved.
	ODFI Id	Number(8)	N/A	8-digit routing number of the ODFI.
	Batch No	Number(7)	N/A	Batch number.
ACH File Control	Section No.	Number(1)	9	Constant number.
	Batch count	Number(6)	N/A	The number of batches sent in the file.
	Block count	Number(6)	N/A	The number of physical blocks in the file, including both File Header and File Control Records. This is the ceiling of the number of records divided by the blocking factor, which is 10.
	Entry count	Number(8)	N/A	The number of entries and addenda in the file.
	Total hash count	Number(10)	N/A	Sum of the Entry Hash fields on the Batch Control Records.
	Total file debit	Number(12)	N/A	Contains the accumulated debit and debit for the file * 10000 (4 implied decimal places).
	Total file credit,	Number(12)	N/A	Contains the accumulated credit and credit for the file * 10000 (4 implied decimal places).
	Reserved	Char(39)	NULL	Reserved.

Table 27–69 (Cont.) Output File

Record Name	Field Name	Field Type	Default Value	Description
ACH Completed Block	End string	Char(94)	N/A	Mark the end of the file: a string of 94 '9' characters. The number of end lines with a string of 94 '9' characters is identified by the following equation: 10 - mod (number of lines in the file, 10).

Design Assumptions

N/A

saexpuar (Export to Universal Account Reconciliation System from Sales Audit)

Module Name	saexpuar.pc
Description	Export to Universal Account Reconciliation System from Sales Audit
Functional Area	Oracle Retail Sales Audit
Module Type	Integration
Module Technology	ProC
Catalog ID	RSA06
Wrapper Script	N/A

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The SAEXPUAR program is used to select the lottery, bank deposit, money order, and credit card totals and write them to output files for export to an external account clearing house application. For each store day, saexpuar posts specified totals to their appropriate output files.

Restart/Recovery

The logical unit of work for this module is defined as a unique store/day combination. Records will be fetched, updated, and inserted in batches of commit_max_ctr. Only two commits will be done. One to establish the store/day lock (this will be done by the package) and the other is done at the end, after a store/day has been completely processed.

I/O Specification

Integration Type Download from Sales Audit

File Name UAR usage type appended with system date.

Integration Contract IntCon000046

Output File Layout

The output file will contain one line for each store/day detail record in a comma-delimited format. The fields are surrounded by double quotes. For example, a record for store 1000 on May 20, 2001 with an amount of 19.99 will look something like this:

"1", "1000", "1999", "20010520","2","","1","","","","","","","","","","MN","RET"

Table 27–70 Output File

Field Name	Field Type	Description
Detail Flag	Char	"1" for detail record.
Store	Number	Store number.
Amount	Number	Total Value * 100 (with 2 implied decimal places).
TranDate	Char	Transaction Date in YYYYMMDD format.
UAR TranCode	Char	Transaction Code. "1" for negative amount, "2" for positive amount.
User Defined Value 1	Char	Ref Number 1 on SA_TOTAL.
User Defined Value 2	Char	Total Seq Number on SA_TOTAL.
User Defined Value 3	Char	Ref Number 2 on SA_TOTAL.
User Defined Value 4	Char	Ref Number 3 on SA_TOTAL.
User Defined Value 5	Char	Not used.
User Defined Value 6	Char	Not used.
User Defined Value 7	Char	Not used.
User Defined Value 8	Char	Not used.
User Defined Value 9	Char	Not used.
User Defined Value 10	Char	Not used.
State	Char	State.
Account	Char	Total Identification on SA_TOTAL.

Design Assumptions

N/A

saprepost (Pre/Post Helper Processes for ReSA Batch Programs)

Module Name saprepost.pc

Description Pre/Post Helper Processes for Sales Audit Batch Programs

Functional Area	Oracle Retail Sales Audit
Module Type	Admin
Module Technology	ProC
Catalog ID	RSA26
Wrapper Script	rmswrap.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

The Sales Audit pre/post module facilitates multi-threading by allowing general system administration functions (such as table deletions or mass updates) to be completed after all threads of a particular Sales Audit program have been processed.

This program will take three parameters: username/password to log in to Oracle, a program before or after which this script must run, and an indicator of whether the script is a pre or post function. It will act as a shell script for running all pre-program and post-program updates and purges.

saprepost contains the following helper functions, which are should be individually scheduled with the related main programs.

Table 27–71 Helper Functions

Catalog ID	Saprepost Job	Related Main Program
RSA47	saprepost saexprms post	saexprms
RSA48	saprepost saexpdw post	saexpdw
RSA39	saprepost saordinvexp post	saordinvexp
RSA51	saprepost saexpsim post	saexpsim
	saprepost sapreexp post	sapreexp

Restart/Recovery

NA

Design Assumptions

N/A

sapurge (Purge Aged Store/Day Transaction, Total Value and Error Data from Sales Audit)

Module Name	sapurge.pc
Description	Purge Aged Store/Day Transaction, Total Value and Error Data from Sales Audit
Functional Area	Oracle Retail Sales Audit
Module Type	Admin

Module Technology	ProC
Catalog ID	RSA21
Wrapper Script	rmswrap_out.ksh

Schedule

Oracle Retail Merchandising Batch Schedule

Design Overview

This program will be run daily to control the size of the tables in the sales audit database. Older information will be deleted to ensure optimal performance of the system as a whole.

Different kinds of data need to be kept in the system for different amounts of time. Transactions, all associated transaction details, and Totals calculated or reported for a store day will be deleted when they meet the following criteria:

- The Business Date for those transactions and totals is older than or equal to today's date minus the days_before_purge parameter set up on the sales audit system parameters.
- No locks exist on the store/day.
- One of the two following statements is true for the store/day:
  - Fully loaded, and all errors either corrected or overridden (sa_store_day.audit_status is A (Audited) and sa_store_day.data_status equals F (Fully loaded)). In addition, there are no outstanding exports (records for the store/day in the sa_export_log table where sa_export_log.status equals R (Ready for export)).
  - Never loaded (sa_store_day.audit_status is U (Unaudited) and sa_store_day.data_status equals R (Ready for import)).

Flash Sales data will be deleted when it meets the following criteria:

- Date is two years before today's date minus the days_before_purge parameter set up on the sales audit system parameters.
- Company open and close dates will also need to be kept for two years plus days_before_purge, so that the historical comparisons in flash sales reporting carry the appropriate weight.

Voucher data will be deleted when it meets the following criteria:

- The redeemed date or the escheat date for the specific voucher type is before today's date minus the purge_no_days on sales audit voucher options table for the corresponding voucher type.

The program can also take in a list of store_day_seq_no to delete. For example, the command line could be: sapurge userid/passwd 1000 1001 1002, where 1000, 1001 and 1003 are store_day_seq_nos that you want to delete. These must also meet the criteria defined above. If a store_day_seq_no is passed to this program, but does not meet the criteria, an error will be written out to the error log.

An output file will be created to store a record for each store and business date that was purged. The file name must be passed in at the command line as a parameter to sapurge.

This program will also purge the data, which is being used for Sales Audit Auditor Framwork and purging criteria based on days_before_purge value from SA_SYSTEM_OPTIONS table.

Restart/Recovery

Restart/recovery is implicit in purge programs. The program only needs to be run again to restart appropriately.

Design Assumptions

N/A

b8d_sa_purge (Purge Into History Tables)

Module Name	b8saprgb.pls/ b8saprgs.pls
Description	Purge records into History tables
Functional Area	Financial data
Module Type	Admin - Ad hoc
Module Technology	Background Processing
Catalog ID	N/A

Design Overview

This background job is composed of two steps processing. It will have a threading assignment and a business logic processing.

Thread assignment program (SA_PURGE_THREAD) will filter eligible records based on sa_store_day, sa_system_options and period tables .These records are chunked and Thread ID is assigned for each. They will be stored temporarily in a staging table b8d_sa_purge_stg table.

The Business logic program (SA_PURGE) will process records from the base tables based on store_day_seq_no, store, and day. Using bulk processing, this program will filter the records from the tables and insert the records into the HIST tables. Then the inserted records will be deleted from the base tables.

The decision to insert or not to insert the records into the HIST tables is based on the archive_ind and archive_job_ind from the b8d_process_config.

1. If the archive_ind ='Y' and archive_job_ind='Y' then the data from the base tables are inserted into the HIST tables.
2. If archive_ind or archive_job_ind is set to 'N'.Then the records are deleted from the base tables without inserting into the HIST tables.

Scheduling Constraints

Schedule Information	Description
Processing Cycle	Ad-hoc
Frequency	Daily - 3 hours interval - 4 times a day

Schedule Information	Description
Scheduling Considerations	N/A
Pre-Processing	N/A
Post-Processing	N/A
Threading Scheme	Threaded by rownum

Restart/Recovery

N/A

Key Tables Affected

Table	Select	Insert	Update	Delete
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
RMS_BATCH_STATUS	Yes	No	No	No
B8D_PROCESS_CONFIG	Yes	No	No	No
B8D_SA_PURGE_STG	No	Yes	No	No
ALL_PART_TABLES	Yes	No	No	No
SA_COMMENTS_HIST	No	Yes	No	No
SA_CUSTOMER_HIST	No	Yes	No	No
SA_CUST_ATTRIB_HIST	No	Yes	No	No
SA_ERROR_HIST	No	Yes	No	No
SA_ERROR_REV_HIST	No	Yes	No	No
SA_EXPORTED_HIST	No	Yes	No	No
SA_EXPORTED_REV_HIST	No	Yes	No	No
SA_EXPORT_LOG_HIST	No	Yes	No	No
SA_FLASH_SALES_HIST	No	Yes	No	No
SA_HQ_VALUE_HIST	No	Yes	No	No
SA_IMPORT_LOG_HIST	No	Yes	No	No
SA_MISSING_TRAN_HIST	No	Yes	No	No
SA_POS_VALUE_HIST	No	Yes	No	No
SA_STORE_DAY_HIST	No	Yes	No	No
SA_STORE_VALUE_HIST	No	Yes	No	No
SA_SYS_VALUE_HIST	No	Yes	No	No
SA_TOTAL_HIST	No	Yes	No	No
SA_TRAN_DISC_HIST	No	Yes	No	No
SA_TRAN_DISC_REV_HIST	No	Yes	No	No
SA_TRAN_HEAD_HIST	No	Yes	No	No
SA_TRAN_HEAD_REV_HIST	No	Yes	No	No

Table	Select	Insert	Update	Delete
SA_TRAN_IGTAX_HIST	No	Yes	No	No
SA_TRAN_IGTAX_REV_HIST	No	Yes	No	No
SA_TRAN_ITEM_HIST	No	Yes	No	No
SA_TRAN_ITEM_REV_HIST	No	Yes	No	No
SA_TRAN_PAYMENT_HIST	No	Yes	No	No
SA_TRAN_PAYMENT_REV_HIST	No	Yes	No	No
SA_TRAN_TAX_HIST	No	Yes	No	No
SA_TRAN_TAX_REV_HIST	No	Yes	No	No
SA_TRAN_TENDER_HIST	No	Yes	No	No
SA_TRAN_TENDER_REV_HIST	No	Yes	No	No
SA_COMMENTS	Yes	No	No	Yes
SA_CUSTOMER	Yes	No	No	Yes
SA_CUST_ATTRIB	Yes	No	No	Yes
SA_ERROR	Yes	No	No	Yes
SA_ERROR_REV	Yes	No	No	Yes
SA_EXPORTED	Yes	No	No	Yes
SA_EXPORTED_REV	Yes	No	No	Yes
SA_EXPORT_LOG	Yes	No	No	Yes
SA_FLASH_SALES	Yes	No	No	Yes
SA_HQ_VALUE	Yes	No	No	Yes
SA_IMPORT_LOG	Yes	No	No	Yes
SA_MISSING_TRAN	Yes	No	No	Yes
SA_POS_VALUE	Yes	No	No	Yes
SA_STORE_DAY	Yes	No	No	Yes
SA_STORE_VALUE	Yes	No	No	Yes
SA_SYS_VALUE	Yes	No	No	Yes
SA_TOTAL	Yes	No	No	Yes
SA_TRAN_DISC	Yes	No	No	Yes
SA_TRAN_DISC_REV	Yes	No	No	Yes
SA_TRAN_HEAD	Yes	No	No	Yes
SA_TRAN_HEAD_REV	Yes	No	No	Yes
SA_TRAN_IGTAX	Yes	No	No	Yes
SA_TRAN_IGTAX_REV	Yes	No	No	Yes
SA_TRAN_ITEM	Yes	No	No	Yes
SA_TRAN_ITEM_REV	Yes	No	No	Yes
SA_TRAN_PAYMENT	Yes	No	No	Yes
SA_TRAN_PAYMENT_REV	Yes	No	No	Yes
SA_TRAN_TAX	Yes	No	No	Yes

Table	Select	Insert	Update	Delete
SA_TRAN_TAX_REV	Yes	No	No	Yes
SA_TRAN_TENDER	Yes	No	No	Yes
SA_TRAN_TENDER_REV	Yes	No	No	Yes
SA_CUSTOMER	Yes	No	No	Yes
SA_POS_VALUE_WKSHT	Yes	No	No	Yes
SA_SYS_VALUE_WKSHT	Yes	No	No	Yes
SA_ERROR_WKSHT	Yes	No	No	Yes
SA_STORE_ACH	Yes	No	No	Yes
SA_ESCHEAT_VOUCHER	Yes	No	No	Yes
SA_ESCHEAT_TOTAL	Yes	No	No	Yes
KEY_MAP_GL	Yes	No	No	Yes
SA_GL_REF_DATA	Yes	No	No	Yes
SA_STORE_DAY_WRITE_LOCK	Yes	No	No	Yes