Retek® Merchandising System 9.0.10

Addendum to Operations Guide

The software described in this documentation is furnished under a license agreement, is the confidential information of Retek Inc., and may be used only in accordance with the terms of the agreement.

No part of this documentation may be reproduced or transmitted in any form or by any means without the express written permission of Retek Inc., Retek on the Mall, 950 Nicollet Mall, Minneapolis, MN 55403, and the copyright notice may not be removed without the consent of Retek Inc.

Information in this documentation is subject to change without notice.

Retek provides product documentation in a read-only-format to ensure content integrity. Retek Customer Support cannot support documentation that has been changed without Retek authorization.

Corporate Headquarters:

Retek Inc.
Retek on the Mall
950 Nicollet Mall
Minneapolis, MN 55403

888.61.RETEK (toll free US)
+1 612 587 5000

European Headquarters:

Retek
110 Wigmore Street
London
W1U 3RW
United Kingdom

Switchboard:
+44 (0)20 7563 4600

Sales Enquiries:
+44 (0)20 7563 46 46
Fax: +44 (0)20 7563 46 10

Printed in the United States of America.

Customer Support

Customer Support hours:

Customer Support is available 7x24x365 via e-mail, phone, and Web access.

Depending on the Support option chosen by a particular client (Standard, Plus, or Premium), the times that certain services are delivered may be restricted. Severity 1 (Critical) issues are addressed on a 7x24 basis and receive continuous attention until resolved, for all clients on active maintenance.

Contact Method	Contact Information
Internet (ROCS)	www.retek.com/support Retek's secure client Web site to update and view issues
E-mail	support@retek.com
Phone	US & Canada: 1-800-61-RETEK (1-800-617-3835) World: +1 612-587-5800 EMEA: 011 44 1223 703 444 Asia Pacific: 61 425 792 927
Mail	Retek Customer Support Retek on the Mall 950 Nicollet Mall Minneapolis, MN 55403

When contacting Customer Support, please provide:

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

Contents

Chapter 1 – Introduction

This addendum to the Retek Merchandising System (RMS) 9.0 Operations Guide contains updates to the following batch designs:

- Forecasting/Replenishment (Fcstprg)
- Clearance Reset Pricing POS Extract (pccrdnld)
- POS Download (posdnld
- RMS to RLS download conversion (rms2rls)
- Unit Quantity (saaldnld.pc)

Refer to the following chapters for that information, which supersedes any comparable information in the RMS 9.0 Operations Guide. Each chapter contains a subsection indicating what specific modifications have been made.

Chapter 2 – Forecasting/Replenishment (Fcstprg)

Design overview

The forecast sales information at the item and hierarchy level is no longer stored on history tables.

In order to speed the performance of the data loads into RMS and subsequent hierarchy forecast builds, the tables have been built either as partitioned views or as Oracle 8 partitioned tables. The data objects are all partitioned by domain.

When new forecasts are created and loaded into RMS from RDF, the entire domain will be re-built and re-loaded. For this reason, a purge program will be used to truncate the partitions that are being re-loaded. This module will take the domain value that is being rebuilt and will truncate the partitions of the tables using dynamic sql.

The client can also choose not to partition the table. If the client built the tables as a partitioned, they will have to set the li_partition_table_ind to 1 manually so that a purge program will be used to truncate the partitions that are being re-loaded.

Scheduling constraints

Pre/Post Logic Description

Processing Cycle: before fcslupld.

Scheduling Diagram:

Pre-Processing:

Post-Processing:

Threading Scheme: threaded by passed parameter.

Restart recovery

Logical Unit of Work (Recommended Commit check points)

Driving Cursor

N/A

No restart/recovery libraries should be included at all.

Program flow

Structure Chart

Shared modules

Listing of all externally referenced functions and Stored procedures and description of usage.

Function level description

All database interactions required and error handling considerations.

Main function

This function will require two input parameters additional to the normal parameter. The domain value, and the table owner will be passed into the function along with the username/password string.

Process()

The truncate_table function should be called 4 times, each time passing the table name to be truncated, along with the table owner name. A character string should be locally declared to hold the table names. The four tables will be named:

item_forecast_X

dept_sales_forecast_X

class_sales_forecast_X

subclass_sales_forecast_X

where X is the domain value parameter.

Truncate_table function.

This function will create a truncate table command as a character string and will use dynamic sql to truncate the table. The truncate table command should indicate which table is to be truncated and the table owner value. Both of these values will be passed as parameters to the program.

I/O specification

All files layouts input and output.

N/A

Chapter 3 – Clearance Reset Pricing POS Extract (pccrdnld)

Design overview

The Clearance Reset Pricing Information (pccrdnld) module is used to send clearance reset pricing information to the point of sale system. When a clearance event is within a pre-determined number of days of its reset date, this program will gather current clearance pricing information and the pricing information to which an item will be reset. This information will be written out to a table that will be used as an interface point with the point of sale. Clearance records will be updated to indicate that they have had reset prices downloaded to the point of sale by setting a reset downloaded date with the current date.

Tables	Index	Select	Insert	Update	Delete
PERIOD	No	Yes	No	No	No
UNIT_OPTIONS	No	Yes	No	No	No
CLEAR_RESET_CALC	No	Yes	No	No	No
CLEAR_SUSP_HEAD	No	Yes	No	No	No
WIN_SKUS	Yes	Yes	No	Yes	No
WIN_STORE	Yes	Yes	No	Yes	No
RAG_STYLE	Yes	Yes	No	Yes	No
RAG_STYLE_ST	Yes	Yes	No	Yes	No
RAG_SKUS_ST	Yes	Yes	No	No	No
POS_MODS	No	No	Yes	No	No
UPC_EAN	Yes	Yes	No	No	No
ITEM_ZONE_PRICE	Yes	Yes	No	No	No

Scheduling constraints

Processing Cycle:	Phase 1
Scheduling Diagram:	N/A
Pre-Processing:	pcdnld should complete processing before this module begins
Post-Processing:	N/A
Threading Scheme:	Clearance

Restart recovery

The driving cursor was changed to get the system_indicator directly using a join to the desc_look table instead of calling the GET_SYS_IND function:

```
EXEC SQL DECLARE c_get_reset CURSOR FOR
    SELECT crc.clearance,
           crc.sku,
           TO_CHAR(crc.reset_date, 'DDMMYYYY'),
           pzgs.store,
           ROWIDTOCHAR(crc.rowid),
           dl.system_ind,
           ';' || TO_CHAR(crc.clearance) ||
           ';' || TO_CHAR(crc.sku) ||
           ';' || TO_CHAR(crc.zone_group_id) ||
           ';' || TO_CHAR(crc.zone_id)
      FROM clear_reset_calc crc,
           clear_susp_head csh,
           price_zone_group_store pzgs,
           desc_look dl,
           v_restart_clearance rv
     WHERE crc.reset_date is NOT NULL
       AND (
              (csh.status = 'C'
               AND EXISTS (SELECT 'x'
                             FROM clear_susp_detail csd
                            WHERE csd.clearance =
csh.clearance
                              AND csd.sku = crc.sku
                              AND csd.sku = dl.sku
                              AND csd.zone_group_id =
crc.zone_group_id
                              AND csd.zone_id =
crc.zone_id
                              AND csd.downloaded_date is
NOT NULL))
            OR
              (csh.status = 'A'
               AND crc.sku = dl.sku
           AND :prc_ext_days >= (crc.reset_date -
```

```
                    to_date(:ps_vdate,'YYYYMMDD'))
              AND crc.reset_date >= TO_DATE(:ps_tomorrow,'YYYYMMDD'))
                )
        AND csh.clearance = crc.clearance
        AND crc.zone_group_id = pzgs.zone_group_id
        AND crc.zone_id = pzgs.zone_id
        AND crc.reset_downloaded_date is NULL
        AND rv.driver_value       = csh.clearance
        AND rv.driver_name        =
:ps_restart_driver_name
        AND rv.num_threads        =
:pi_restart_num_threads
        AND rv.thread_val         =
:pi_restart_thread_val
    UNION
    SELECT crc.clearance,
           crc.sku,
           crc.zone_group_id,
           crc.zone_id,
           TO_CHAR(crc.reset_date, 'YYYYMMDD'),
           pbt.store,
           ROWIDTOCHAR(crc.rowid),
           dl.system_ind,
           ';' || TO_CHAR(crc.clearance) ||
           ';' || TO_CHAR(crc.sku) ||
           ';' || TO_CHAR(crc.zone_group_id) ||
           ';' || TO_CHAR(crc.zone_id)
      FROM clear_reset_calc crc,
           clear_susp_head csh,
           price_batch_tran pbt,
           desc_look dl,
           v_restart_clearance rv
     WHERE crc.reset_date is NOT NULL
       AND (
              (csh.status = 'C'
               AND EXISTS (SELECT 'x'
                           FROM clear_susp_detail csd
                           WHERE csd.clearance =
csh.clearance
```

```
                                AND csd.sku = crc.sku
                                AND csd.zone_group_id =
crc.zone_group_id
                                AND csd.zone_id =
crc.zone_id
                                AND csd.downloaded_date is
NOT NULL))
            OR
              (csh.status = 'A'
               AND :prc_ext_days >= (crc.reset_date -
to_date(:ps_vdate,'YYYYMMDD'))
               AND crc.reset_date >=
TO_DATE(:ps_tomorrow,'YYYYMMDD'))
            )
       AND csh.clearance = crc.clearance
       AND crc.sku = dl.sku
           AND pbt.zone_group_id   = crc.zone_group_id
           AND pbt.new_zone_id     = crc.zone_id
           AND pbt.effective_date  <
(TO_DATE(:ps_tomorrow,'YYYYMMDD')
                                                  +
:prc_ext_days - 1)
       AND crc.reset_downloaded_date is NULL
       AND rv.driver_value       = csh.clearance
       AND rv.driver_name        =
:ps_restart_driver_name
       AND rv.num_threads        =
:pi_restart_num_threads
       AND rv.thread_val         =
:pi_restart_thread_val
  ORDER BY 1, 2, 3, 4;
```

Program flow

N/A

Shared modules

N/A

Function level description

The call to the get_sys_ind function is removed and the system indicator is fetched back in the driving cursor.

I/O specification

N/A

Technical issues

N/A

Chapter 4 – POS Download (posdnld)

Design overview

The posdnld program is used to download pos_mods records created in the RMS to the store POS systems. This program has one output file which contains all records for all stores in a given run.

Scheduling constraints

Processing Cycle: PHASE 4 (daily)

Scheduling Diagram: This program is run towards the end of the batch run when all pos_mods records have been created for the transaction day.

Pre-Processing: N/A

Post-Processing: N/A

Threading Scheme: N/A

Restart recovery

Restart/recovery for this program is set up at the store/upc or sku level. Threading is done by store using the v_restart_store 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 recommended commit counter setting is 10000 records (subject to change based on experimentation).

Program flow

Shared modules

N/A

Function level description

Init

This function initializes restart/recovery for this program. It also opens the output file, retrieves system variables and calls a function to size the arrays used in this program.

Process

This function drives the processing of the program. The driving cursor is fetched here which retrieves all the records from pos_mods where the pos_mods.store value is greater than zero. Once the records are fetched, the write_rec() function is called to perform processing on them. Restart/Recovery and commiting of records is also performed here.

Write_rec

This function will prepare records for insert into the output file. This program used the Retek standard file format FHEAD, FDETL, FTAIL.

Final

This function will finish restart/recovery logic, close the output file and delete the temporary output file used while the program processes.

Init_format_strs

This function formats the strings for the FHEAD, FDETL, and FTAIL records in the output file.

Init_arrays

This function initializes the size of the array used for the driving cursor fetch the size of the restart max counter on restart_control.

Resize_arrays

This function increases the memory for the driving cursor array by the size of the restart max counter on restart_control.

Get_upc_type

This function query the upc_ean table to get the var_type given the passed in upc and upc_supp.

I/O specification

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 Identifier	Number(10)	Sequenctial number Created by program.	ID of current line being created for output file.
	File Type Definition	Char(4)	POSD	Identifies file as 'POS Download'
	File Create Date	Char(14)	create date	current date, formatted to 'YYYYMMDDHH24MISS'.
File Detail	File Type Record Descriptor	Char(5)	FDETL	Identifies file record type
	File Line Identifier	Number(10)	Sequenctial number Created by program.	ID of current line being created for output file.
	Location Number	Number(4)	store identifier	Store identifier from the store table in Retek
	Update Type	Char(1)	update type	Code used for client specific POS system.
	Start_Date	Char(14)	start date	date for the change to take effect at the POS, formatted to 'YYYYMMDDHH24MISS'.
	Upc	Number(13)	upc identifier	the id number of a UPC. From the Retek upc_ean table.

Record Name	Field Name	Field Type	Default Value	Description
	Upc Supplement	Number(5)	supplemental identifier	used to further specify the id of an UPC item. From the Retek upc_ean table.
	Upc var type	Char(1)	Variable upc type indicator	Identifies what type of variable UPC is being sent. Valid values are 'W'eight, 'P'rice, and NULL.
	Tran Type	Number(2)	transaction type	the transaction type for the record from the Retek pos_mods table.
	SKU	Number(8)	SKU identifier	the id number of a SKU from the Retek desc_look table.
	SKU Description	Char(40)	SKU description	the description of the SKU from the Retek desc_look table.
	Dept	Number(4)	dept id	the id of the dept for the item from the Retek win_skus or rag_style table.
	Class	Number(4)	class id	the id of the class for the item from the Retek win_skus or rag_style table.
	Subclass	Number(4)	subclass id	the id of the subclass for the item from the Retek win_skus or rag_style table.
	New Price	Number(20)	new price	the new price to be taken at the POS. This value is from the Retek pos_mods table.

Record Name	Field Name	Field Type	Default Value	Description
	Multi Units	Number(12)	multi units	Number of multi units
	Multi Units Retail	Number(20)	multi units retail	unit retail for the multi units
	Status	Char(1)	status	Populates if tran_type for the item is 1(new item added) or 25 (change item status) or 26 (change taxable indicator).
	Taxable Indicator	Char(1)	taxable ind	Populates if tran_type for the item is 1(new item added) or 25 (change item status) or 26 (change taxable indicator).
	Promotion Number	Number(4)	promotion number	Promotion number for SKU. This value is from the Retek system.
	Mix Match Number	Number(4)	mix match number	mix match number for SKU. This value is from the Retek ssytem.
	Mix Match Type	Char(1)	mix match type	mix match type (Buy or Get) for SKU. This value is from the Retek system.
	Threshold Number	Number(4)	threshold number	Threshold number for SKU. This value is from the Retek system.
File Trailer	File Type Record Descriptor	Char(5)	FTAIL	Identifies file record type

Record Name	Field Name	Field Type	Default Value	Description
	File Line Identifier	Number(10)	Sequential number Created by program.	ID of current line being created for output file.
	File Record Counter	Number(10)		Number of records/transactions processed in current file (only records between head & tail)

Technical issues

N/A

Chapter 5 – RMS to RLS download conversion (rms2rls)

Design overview

This NAWK program will convert Retek generic download files to the format specified by the Nautilus' download file. This involves removing Retek standard flat file specifications, such as FHEAD, FDETL, and FTAIL records, and separating the file by warehouse, so that each specific warehouse will have its own file.

See below for the download programs that produce files to be converted with rms2rls.awk.

Scheduling constraints

Processing Cycle: N/A

Scheduling Diagram: After processing of download programs that produce standard Retek flat files, each file needs to be run through the conversion program before Nautilus download programs can run.

Pre-Processing: N/A

Post-Processing: Nawk program will create a .logfile, which will record the success or failure of the conversion process.

Threading Scheme: N/A

Restart recovery

This is a flat file conversion program done from the Unix shell in an AWK script, no restart/recovery, commit is used or practical.

If there are conversion errors, redo the whole conversion.

Program flow

N/A

Function level description

Read in the first record of the input file (ie. FHEAD).

Assign output file names to variable, OUTFILE, depending upon the file type of the input file located in the FHEAD record.

Process the details of the input file by removing the Retek standard file fields, such as FDETL, THEAD, and TDETL, and writing the record to the correct output file.

Append the DC number to the output file name. Since there may be multiple warehouses processed in one input file, there will be one output file per warehouse.

Write the detail records to the appropriate output file. Some input files will be parsed into several output files, depending upon the Nautilus specified file layout.

Record any errors to the .logfile file.

I/O specification

Input file: any file produces by the Retek download programs. These files are specified on the command line. The appropriate output file will be produced as specified by the Nautilus download batch programs.

File type	Rms2Rls.awk Output File	Download Program Name
ASND	asn_header.datDC#	asndnld.pc
ASND	asn_container.datDC#	asndnld.pc
ASND	asn_container_item.datDC#	asndnld.pc
ASND	asn_item.datDC#	asndnld.pc
CPTT	component_tkt.datDC#	Allocupd.pc
CPTT	component_tkt.datDC#	asndnld.pc
CPTT	component_tkt.datDC#	ordrev.pc
CPTT	component_tkt.datDC#	tsfdnld.pc
BOMD	bill_of_materials.dat	itemdnld.pc
ITEM	item.dat	itemdnld.pc
IWOD	inbound_work_order.datDC#	powodld.pc
LOCD	ship_dest.dat	locdnld.pc
OWOD	outbound_work_order.dateDC#	tsfdnld.pc
POHD	po_header.datDC#	ordrev.pc
PODT	po_detail.datDC#	ordrev.pc
POHD	po_header.datDC#	ordprg.pc
PRHD	rcv_pre_dist_header.datDC#	ordrev.pc
PRDT	rcv_pre_dist_detail.datDC#	ordrev.pc
PRDT	rcv_pre_dist_detail.datDC#	allocupd.pc
STOR	rcv_stock_order_header.datDC#	allocupd.pc
STOR	rcv_stock_order_header.datDC#	ordrev.pc
STOR	rcv_stock_order_header.datDC#	saaldnld.pc
STAL	rcv_stock_order_detail.datDC#	allocupd.pc
STAL	rcv_stock_order_detail.datDC#	ordrev.pc
STAL	rcv_stock_order_detail.datDC#	saaldnld.pc

File type	Rms2Rls.awk Output File	Download Program Name
TDCH	po_header.datDC#	tsfdcdld.pc
TDCD	po_detail.datDC#	tsfdcdld.pc
TDCH	po_header.datDC#	tsfprg.pc
TDCH	po_header.datDC#	tsfalprg.pc
TSFD	rcv_stock_order_header.datDC#	tsfdnld.pc
TSFD	rcv_stock_order_detail.datDC#	tsfdnld.pc
UPCD	upc.dat	itemdnld.pc
VNDD	vendor.dat	supdnld.pc

Technical issues

N/A

Chapter 6 – Unit Quantity (saaldnld.pc)

Design overview

The saaldnld.pc needs to be modified to change the format of Unit Quantity, when downloading information to RDM. The new Unit Quantity field for the interface is now Number(12,4) opposed to the original Number(6).

Scheduling constraints

Pre/Post Logic Description.

No change to RMS 8.0.

Restart recovery

Logical Unit of Work (recommended Commit check points)

Driving Cursor

No change.

Program flow

Structure Chart

No change.

Shared modules

Listing of all externally referenced functions and Stored procedures and description of usage.

No change.

Function level description

All database interactions required and error handling considerations.

The saaldnld.pc needs to be modified to change the format of Unit Quantity, when downloading information to RDM. The new Unit Quantity field for the interface is now Number(12,4) opposed to the original Number(6).

I/O specification

All files layouts input and output

Stock order file

File Header:

Field Name	Default Value	Field Type	Description
Record Type	FHEAD	Char(5)	Identifies file record type
Line Number	Incremented internally	Number(10)	Sequential line number
File Type	STOR	Char(5)	Identifies file
Create Date	YYYYMMDD DHH24MISS format	Char(14)	File create date

File Detail:

Field Name	Default Value	Field Type	Description
Record type	FDETL	Char(5)	identifies the Detail line
Line number	Incremented internally	Number(10)	Sequential line number
Action_type	‘A’, ‘M’, ‘D’	Char(1)	Add, modify, or delete action type
Location	Alloc_header.wh	Number(4)	From Warehouse location
Transaction Date/Time	Sysdate	Datetime(12)	System date
Distribution Number	‘A’ + alloc_no	Char(9) or char(11) if the allocation_ind is = ‘Y’.	Allocation number. Prefix ‘A’ for alloc
Download Comment	NULL	Char(30)	Comment to be printed on the label (future use)

Field Name	Default Value	Field Type	Description
Pick-Not-Before Date	Alloc_header. release_date or vdate if NULL release_date	Char(8)	Date before which merchandise will not be distributed
Pick-Not-After Date	PNB date + (default period from codes table where code type = 'DEFT' and code = 'DATE')	Char(8)	Date by which merchandise must be distributed
Event code	NULL	Char(6)	Identifier of event
Event description	NULL	Char(25)	Event description
Priority	1	Char(4)	Priority
Order Type	UPPER(system _options.default _order_type)	Char(9)	Type of Order: AUTOMATIC, MANUAL or WAVE
Break By Distro	'N'	Char(1)	Contols the mixing of orders (distros) in a container
Carrier Code	NULL	Char(4)	Code of the carrier for the order
Carrier Service Code	NULL	Char(6)	Carrier's service code for the delivery, First Class, and son on (future Use)
Route	NULL	Char(10)	Route specified for the delievery
Ship Address Description	NULL	Char(30)	The description (such as the store name). This is the first line of the address block

Field Name	Default Value	Field Type	Description
Ship Address Line 1	NULL	Char(30)	Shipping Address Line 1
Ship Address Line 2	NULL	Char(30)	Shipping Address Line 2
Ship Address Line 3	NULL	Char(30)	Shipping Address Line 3
Ship Address Line 4	NULL	Char(30)	Shipping Address Line 4
Ship Address Line 5	NULL	Char(30)	Shipping Address Line 5
City	NULL	Char(25)	Shipping City
State	NULL	Char(3)	Shipping State
Zip	NULL	Char(10)	Shipping Zip
Billing Address Description	NULL	Char(30)	The Description (such as company name). This is the first line of the address block.
Billing Address Line 1	NULL	Char(30)	Billing Address Line 1
Billing Address Line 2	NULL	Char(30)	Billing Address Line 2
Billing Address Line 3	NULL	Char(30)	Billing Address Line 3
Billing Address Line 4	NULL	Char(30)	Billing Address Line 4
Billing Address Line 5	NULL	Char(30)	Billing Address Line 5
Amount 1	0.00	Number(8,2)	Amount Charge 1
Amount 2	0.00	Number(8,2)	Amount Charge 2
Amount 3	0.00	Number(8,2)	Amount Charge 3

Field Name	Default Value	Field Type	Description
Po number	NULL	Char(9)	Unique identifier of the purchase order, prefixed with 'P'

File Trailer:

Field Name	Default Value	Field Type	Description
Record Type	FTAIL	Char(5)	Identifies file record type
Line Number	Incremented internally	Number(10)	Sequential line number
File Line Count	Total number of detail lines	Number(6)	Total number of transaction lines in file (not including FHEAD and FTAIL)

Stock Allocation file

File Header:

Field Name	Default Value	Field Type	Description
Record Type	FHEAD	Char(5)	Identifies file record type
Line Number	Incremented internally	Number(10)	Sequential line number
File Type	STAL	Char(5)	Identifies file
Create Date	YYYYMMDD HH24MISS format	Char(14)	File create date

File Detail:

Field Name	Default Value	Field Type	Description
Record Type	FDETL	Char(5)	identifies the Detail line
Line Number	Incremented internally	Number(10)	Sequential line number
Action_type	'A', 'M', 'D'	Char(1)	Add, modify, or delete action type
Location	Alloc_header.wh	Number(4)	From Warehouse location
Transaction Date/Time	Sysdate	Datetime(12)	System date
Distribution Number	'A' + alloc_no	Char(9) or char(11) if the allocation_ind is = 'Y'.	Allocation number. Prefix 'A' for alloc
Item ID	Alloc_header.sku	Char(16)	Unique item identifier
Requested Unit Quantity	Alloc_detail.qty_allocated	Number(12, 4)	Quantity allocated
Destination ID	Alloc_detail.store or wh	Number(4)	Allocation location
Price	Item_zone_price.unit_retail	Number(5,2)	Retail price
Print UPC Flag	'N'	Char(1)	Print upc flag (future use)
Ticket Type	Item_ticket.ticket_type_id or '0000'	Char(4)	Receiving Ticket type of item.
Priority	'1'	Char(4)	Priority (highest 1)
Expedite Flag	'N'	Char(1)	Flag indicating whether the order should be shipped via normal or expedite carrier service

File Trailer:

Field Name	Default Value	Field Type	Description
Record Type	FTAIL	Char(5)	Identifies file record type
Line Number	Incremented internally	Number(10)	Sequential line number
File Line Count	Total number of detail lines	Number(6)	Total number of transaction lines in file (not including FHEAD and FTAIL)

Component ticketing file

File Header:

Field Name	Default Value	Field Type	Description
Record Type	FHEAD	Char(5)	Identifies file record type
Line Number	Incremented internally	Number(10)	Sequential line number
File Type	CPTT	Char(5)	Identifies file
Create Date	YYYYMMDD HH24MISS format	Char(14)	File create date

File Detail:

Field Name	Default Value	Field Type	Description
Record type	FDETL	Char(5)	identifies the Detail line
Line number	Incremented internally	Number(10)	Sequential line number
Action_type	'A'	Char(1)	Add, modify, or delete action type
Location	Alloc_header.wh	Number(4)	From Warehouse location
Transaction Date/Time	Sysdate	Datetime(12)	System date

Field Name	Default Value	Field Type	Description
Distribution Number	'A' + alloc_no	Char(9) or char(11) if the allocation_ind is = 'Y'.	Allocation number. Prefix 'A' for alloc
Item ID	Alloc_header.sku	Char(16)	Unique item identifier
Destination ID	Alloc_detail.store or wh	Number(4)	Allocation location
Component Item ID	v_packsku_qty.sku	Char (16)	Item identifier of the component
Price	Item_zone_price.unit_retail	Number(5,2)	Retail price

File Trailer:

Field Name	Default Value	Field Type	Description
Record Type	FTAIL	Char(5)	Identifies file record type
Line Number	Incremented internally	Number(10)	Sequential line number
File Line Count	Total number of detail lines	Number(6)	Total number of transaction lines in file (not including FHEAD and FTAIL)