

**Oracle® Retail Merchandising System  
Operations Guide Addendum  
Release 11.0.7  
March 2006**

Copyright © 2006, Oracle. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

**U.S. GOVERNMENT RIGHTS** Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software—Restricted Rights (June 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

# Contents

<b>Preface.....</b>	<b>v</b>
Audience .....	v
Related Documents .....	v
Customer Support .....	v
<b>1 Stock Count Stock on Hand Updates [stkvar] .....</b>	<b>7</b>
Design Overview .....	7
Input Specifications .....	8
Scheduling Constraints .....	9
Restart Recovery.....	9
Shared Modules .....	9
<b>2 Stock Count Explode [stkxpld].....</b>	<b>11</b>
Design Overview .....	11
Program Flow .....	12
Shared Modules .....	12
Function Level Description.....	12
Input Specification.....	13
Scheduling Constraints .....	13
Restart Recovery.....	13
Technical Issues.....	13



---

# Preface

Oracle Retail Operations Guides are designed so that you can view and understand the application's 'behind-the-scenes' processing.

## Audience

Anyone with an interest in developing a deeper understanding of the underlying processes and architecture supporting Oracle Retail Merchandising System (RMS) functionality will find valuable information in this guide. There are three audiences in general for whom this guide is written:

- Business analysts looking for information about processes and interfaces to validate the support for business scenarios within and other systems across the enterprise.
- System analysts and system operations personnel:
  - Who are looking for information about RMS's processes internally or in relation to the systems across the enterprise.
  - Who operate RMS regularly.
- Integrators and implementation staff with overall responsibility for implementing RMS.

## Related Documents

You can find more information about this product in these resources:

- Oracle Retail Merchandising System Installation Guide
- Oracle Retail Merchandising System Release Notes
- Oracle Retail Merchandising System Data Model

## Customer Support

- <https://metalink.oracle.com>

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.



## Stock Count Stock on Hand Updates [stkvar]

### Design Overview

The stkvar.pc program updates the stock on hand and computes the total cost and total retail in the stake\_prod\_loc.

Tables Affected:

TABLE	INDEX	SELECT	INSERT	UPDATE	DELETE
CLASS	No	Yes	No	No	No
ITEM_LOC	Yes	No	No	Yes	No
ITEM_LOC_SOH	Yes	No	No	Yes	No
ITEM_MASTER	No	Yes	No	No	No
ITEM_SUPP_COUNTRY	No	Yes	No	No	No
ITEM_XFORM_DETAIL	No	Yes	No	No	No
ITEM_XFORM_HEAD	No	Yes	No	No	No
NWP	No	No	Yes	Yes	No
NWP_FREEZE_DATE	No	Yes	No	No	No
STAKE_CONT	No	Yes	No	No	Yes
STAKE_HEAD	No	Yes	No	No	No
STAKE_PROD_LOC	Yes	No	No	Yes	No
STAKE_QTY	No	Yes	No	No	No
STAKE_SKU_LOC	No	Yes	No	Yes	No
WH	No	Yes	No	No	No

Indexes:

STAKE\_PROD\_LOC (dept, store, wh, data\_type)

This program updates the stock on hand for all items as a result of a stock take.

The program is driven by STAKE\_CONT, in conjunction with STAKE\_SKU\_LOC where the ITEM, loc and cycle count on STAKE\_SKU\_LOC match those on STAKE\_CONT, and where STAKE\_CONT run\_type = 'A' (for adjustment).

For each row retrieved from the above tables, the unit systems are processed as follows:

- An ITEM\_LOC\_SOH record is updated for every ITEM/loc combination. The new stock on hand = item\_loc\_soh.stock\_on\_hand - snapshot\_stock\_on\_hand\_qty (from the STAKE\_SKU\_LOC table) + the physical count quantity on STAKE\_SKU\_LOC. In addition, the pack\_comp\_soh field is updated on ITEM\_LOC when a pack is processed for each component ITEM in the pack.
- Total cost and total retail are computed as the snapshot unit retail times the sum of the physical count quantity plus the snapshot in-transit (from the STAKE\_SKU\_LOC table).
- STAKE\_PROD\_LOC total cost and total retail amounts are updated with the total cost and total retail for each department, class, subclass, location combination that exists on the cycle count. A record for each is added. If a record already exists on the table, the total cost or retail amount value is adjusted to be the existing total cost or retail amount + the cycle count cost or retail. If no record exists, a new one is added to the table with the value of total cycle count cost or retail for the total cost or retail amount. If the stock ledger is designated not to include VAT on the SYSTEM\_OPTIONS table, the total retail amount will have any VAT amount stripped from it.

Re-run:

- If this program terminates normally, ITEM\_LOC, STAKE\_QTY,
- ITEM\_LOC\_SOH, STAKE\_PROD\_LOC and STAKE\_CONT must be recovered prior to restart. If this program terminates abnormally, restart without recovery.

The syntax for invoking this program is:

stkvar userid/pswd [report\_name ].

Here are some examples:

stkvar userid/pswd( it will not produce any report.)

stkvar userid/pswdany.rpt ( it will produce a report, any.rpt. )

## Input Specifications

```
EXEC SQL DECLARE c_item CURSOR FOR
  SELECT /* ORDERED USE_HASH(stake_cont) FULL(stake_head) */
    sc.item,
    sc.loc_type,
    sc.location,
    c.class_vat_ind,
    ROWIDTOCHAR(sc.ROWID),
    ssl.snapshot_on_hand_qty,
    NVL(ssl.snapshot_in_transit_qty,0),
    NVL(ssl.snapshot_unit_cost, 0),
    NVL(ssl.snapshot_unit_retail, 0),
    NVL(ssl.physical_count_qty, 0),
    ssl.pack_comp_qty,
    NVL(ssl.dept, 0),
    NVL(ssl.class, 0),
    NVL(ssl.subclass, 0),
    ROWIDTOCHAR(ssl.ROWID),
    im.pack_ind,
    TO_CHAR(sh.stocktake_date,'YYYYMMDD'),
    sh.stocktake_type,
    sh.cycle_count,
    ssl.xform_item_type,
    im.deposit_item_type,
    ';' || sc.item ||
```

```

      ';' || TO_CHAR(sc.loc_type) ||
      ';' || TO_CHAR(sc.location)
  FROM stake_sku_loc ssl,
       stake_cont sc,
       stake_head sh,
       item_master im,
       class c,
       wh w,
       v_restart_all_locations rv
 WHERE sc.run_type          = 'A'
   AND sc.item              = ssl.item
   AND sc.loc_type          = ssl.loc_type
   AND sc.location          = ssl.location
   AND sc.cycle_count       = ssl.cycle_count
   AND sh.cycle_count       = ssl.cycle_count
   AND NVL(ssl.xform_item_type, ' ') <> 'S'/*exclude sellable only items*/
   AND im.item              = sc.item
   AND im.item_level        = im.tran_level
   AND im.dept              = c.dept
   AND im.class             = c.class
   AND ssl.location         = w.wh (+)
   AND (im.pack_ind         = 'N' OR
        (ssl.loc_type        = 'W' AND
         im.pack_ind         = 'Y' AND
         w.finisher_ind     = 'N'))
   AND rv.driver_value      = sc.location
   AND rv.driver_name       = :ora_restart_driver_name
   AND rv.num_threads       = TO_NUMBER(:ora_restart_num_threads)
   AND rv.thread_val        = TO_NUMBER(:ora_restart_thread_val)
   AND (sc.item             > NVL(:ora_restart_item,-999) OR
        (sc.item             = :ora_restart_item AND
         (sc.loc_type        > TO_CHAR(:ora_restart_loc_type) OR
          (sc.loc_type        = TO_CHAR(:ora_restart_loc_type) AND
           (sc.location        > TO_NUMBER(:ora_restart_location))))))
        )
      )
  ORDER BY sc.location,
           sc.loc_type,
           sc.item;

```

## Scheduling Constraints

Processing Cycle: PHASE 3  
 Scheduling Diagram: N/A  
 Pre-Processing: N/A  
 Post-Processing: N/A  
 Threading Scheme: LOCATION

## Restart Recovery

This program will be threaded by department and will utilize restart/recovery logic based on item/loc\_type/location.

## Shared Modules

PRICING\_ATTRIB\_SQL.GET\_RETAIL



## Stock Count Explode [stkxpld]

### Design Overview

Stock Count Explode to Item/Location for Stores/Warehouses/External Finishers. This program combined stkxplst and stkxplwh to create a new program [stkxpld]. The Stock Count Explode batch program sets up the stock count snapshot by populating the stocktake tables with department-class-subclass relationship for all items to be counted for a chosen location.

Tables Affected:

TABLE	INDEX	SELECT	INSERT	UPDATE	DELETE
ITEM_LOC	No	Yes	No	No	No
ITEM_MASTER	No	Yes	No	No	No
ITEM_XFORM_DETAIL	No	Yes	No	No	No
ITEM_XFORM_HEAD	No	Yes	No	No	No
PERIOD	No	Yes	No	No	No
STAKE_HEAD	Yes	Yes	No	No	No
STAKE_LOCATION	Yes	Yes	No	No	No
STAKE_PROD_LOC	Yes	No	Yes	No	No
STAKE_PRODUCT	Yes	Yes	No	No	No
STAKE_SKU_LOC	Yes	No	Yes	No	No
SUBCLASS	No	Yes	No	No	No
SYSTEM_OPTIONS	No	Yes	No	No	No

This program inserts into stake\_sku\_loc and stake\_prod\_loc tables for all valid item/loc and department/loc combinations for unit and dollar type stock counts, based on location, department, class and subclass information stored on stake\_location and stake\_product. The stock count can be requested at department, class or subclass level and exploded to Item/loc from any of these levels. Inserts are made into stake\_prod\_loc now at subclass/loc level. This program also inserts into stake\_sku\_loc and deletes stake\_product for all unit only counts for warehouses. Orderable only transformed item, an indicator 'O' is inserted in xform\_item\_type column of stake\_sku\_loc table. Also, for sellable only transformed item, an indicator 'S' is used.

## **Program Flow**

N/A

## **Shared Modules**

N/A

## **Function Level Description**

main()

- Standard Retek main function
- Validates input parameters
- Calls init, process and final
- Logs appropriate message

init()

- Initialize restart variables
- Collect system date information from PERIOD and SYSTEM\_OPTIONS

process()

- The cycle count data with stocktake date equal to the threshold date, which is equal to the vdate plus the stake lockout days, should exist both in STAKE\_HEAD and STAKE\_LOCATION tables
- Open the product cursor
- process\_product function callout for each data to be processed
- Fetch the product cursor into global variables
- if NO\_DATA\_FOUND in the cursor, break the loop

process\_product()

- Insert processed data into STAKE\_SKU\_LOC
- For unit and dollar type of stock count ('B'), insert data into STAKE\_PROD\_LOC

final()

- Restart/recovery close, close files

## Input Specification

```

EXEC SQL DECLARE C_product CURSOR FOR
  SELECT h.cycle_count,
         h.loc_type,
         h.stocktake_type,
         l.location
    FROM stake_location l,
         stake_head h,
         v_restart_all_locations rv
   WHERE h.cycle_count = l.cycle_count
     AND h.stocktake_date = (TO_DATE(:threshold_date, 'DDMMYYYY'))
     AND h.delete_ind = 'N'
     AND rv.driver_value = l.location
     AND rv.num_threads = TO_NUMBER(:ps_num_threads)
     AND rv.thread_val = TO_NUMBER(:ps_thread_val)
     AND (h.cycle_count > NVL(TO_NUMBER(:ps_restart_cycle_count), -999) OR
          (h.cycle_count = TO_NUMBER(:ps_restart_cycle_count) AND
           (l.location > TO_NUMBER(:ps_restart_location) )))
  ORDER BY h.cycle_count,
           l.location;

```

## Scheduling Constraints

Processing Cycle: PHASE 3  
 Scheduling Diagram: N/A  
 Pre-Processing: N/A  
 Post-Processing: STKUPD.PC  
 Threading Scheme: LOCATION

## Restart Recovery

This program will be threaded by location and will utilize restart/recovery logic based on cycle\_count/location

## Technical Issues

N/A