Retek® Merchandising System™ 9.0.14

Operations Guide Addendum

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

Fax: +1 612.587.5100

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. Retek customers on active maintenance agreements may contact a global Customer Support representative in accordance with contract terms in one of the following ways.

Contact Method	**Contact Information**
E-mail	support@retek.com
Internet (ROCS)	rocs.retek.com Retek's secure client Web site to update and view issues
Phone	1 612 587 5800

Toll free alternatives are also available in various regions of the world:

Australia	1 800 555 923 (AU-Telstra) or 1 800 000 562 (AU-Optus)
France	0800 90 91 66
United Kingdom	0800 917 2863
United States	1 800 61 RETEK or 800 617 3835

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

ifdaydnld.pc

Functional Area

RDF Interfaces

Module Affected

ifdaydnld.pc (new)

Design Overview

This module will query the store sales and warehouse issues histories for forecastable items as frequently as once a day. Four flat files for use in RDF modules will be created based on this data: one each for regular, promotion and clearance sales, and a single file for warehouse issues.

Stored Procedures / Shared Modules (Maintainability)

NA

Input Specifications

'Table-To-Table'

This program contains six distinct driving cursors, three for store sales (one each at department, class, and subclass domain levels), and three for warehouse issues (one each at department, class, and subclass domain levels). Only one cursor from each group of three (based on the system_options.domain_level value) will actually be used in a given instance of the program.

Store Sales Driving Cursors:

If the domain_type = 'D', then the driving cursor is as follows:

```
          SELECT itd.store,
                 itd.sku,
                 itd.tran_date,
                 SUM(itd.units),
                 itd.sales_type
            FROM if_tran_data itd,
                 win_skus wsk,
                 win_store wst,
                 domain_dept dd
           WHERE wsk.forecast_ind = 'Y'
             AND itd.tran_code    = 1   /* sales tran_code
*/
```

```
          AND itd.store        = wst.store
          AND itd.sku          = wsk.sku
          AND itd.sku          = wst.sku
          AND ((itd.tran_date >
NVL(wst.last_sales_export_date, itd.tran_date - 1)
                   AND dd.load_sales_ind = 'N')
               OR (dd.load_sales_ind = 'Y'))
          AND wsk.dept         = dd.dept
          AND dd.domain_id     = :pi_restart_thread_val
          AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                (itd.sku  =  :ps_restart_sku AND
                 itd.store > :ps_restart_location))
        GROUP BY itd.store,
                 itd.sku,
                 itd.tran_date,
                 itd.sales_type

       UNION ALL

       SELECT itd.store,
              itd.sku,
              itd.tran_date,
              SUM(itd.units),
              itd.sales_type
         FROM if_tran_data itd,
              rag_skus_st rss,
              rag_style rs,
              domain_dept dd
        WHERE rs.forecast_ind = 'Y'
          AND itd.tran_code   = 1
          AND itd.store       = rss.store
          AND rss.style       = rs.style
          AND itd.sku         = rss.sku
          AND ((itd.tran_date >
NVL(rss.last_sales_export_date, itd.tran_date - 1)
                   AND dd.load_sales_ind = 'N')
               OR (dd.load_sales_ind = 'Y'))
          AND rs.dept         = dd.dept
```

```
          AND dd.domain_id    = :pi_restart_thread_val
          AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                (itd.sku   = :ps_restart_sku AND
                 itd.store > :ps_restart_location))
        GROUP BY itd.store,
                 itd.sku,
                 itd.tran_date,
                 itd.sales_type
       ORDER BY 2, 1;  /* sku, store */
```

If the domain_type = ‘C’, then the driving cursor is as follows:

```
       SELECT itd.store,
              itd.sku,
              itd.tran_date,
              SUM(itd.units),
              itd.sales_type
         FROM if_tran_data itd,
              win_skus wsk,
              win_store wst,
              domain_class dc
        WHERE wsk.forecast_ind = 'Y'
          AND itd.tran_code    = 1   /* sales tran_code
*/
          AND itd.store        = wst.store
          AND itd.sku          = wsk.sku
          AND itd.sku          = wst.sku
          AND ((itd.tran_date >
NVL(wst.last_sales_export_date, itd.tran_date - 1)
                   AND dc.load_sales_ind = 'N')
               OR (dc.load_sales_ind = 'Y'))
          AND wsk.dept         = dc.dept
          AND wsk.class        = dc.class
          AND dc.domain_id     = :pi_restart_thread_val
          AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                (itd.sku  =  :ps_restart_sku AND
                 itd.store > :ps_restart_location))
        GROUP BY itd.store,
                 itd.sku,
                 itd.tran_date,
```

```
                itd.sales_type

        UNION ALL

        SELECT itd.store,
               itd.sku,
               itd.tran_date,
               SUM(itd.units),
               itd.sales_type
          FROM if_tran_data itd,
               rag_skus_st rss,
               rag_style rs,
               domain_class dc
         WHERE rs.forecast_ind = 'Y'
           AND itd.tran_code   = 1
           AND itd.store       = rss.store
           AND rss.style       = rs.style
           AND itd.sku         = rss.sku
           AND ((itd.tran_date >
NVL(rss.last_sales_export_date, itd.tran_date - 1)
                 AND dc.load_sales_ind = 'N')
              OR (dc.load_sales_ind = 'Y'))
           AND rs.dept         = dc.dept
           AND rs.class        = dc.class
           AND dc.domain_id    = :pi_restart_thread_val
           AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                (itd.sku   = :ps_restart_sku AND
                 itd.store > :ps_restart_location))
         GROUP BY itd.store,
                  itd.sku,
                  itd.tran_date,
                  itd.sales_type
        ORDER BY 2, 1;  /* sku, store */
```

If the domain_type = ‘S’, the driving cursor is as follows:

```
        SELECT itd.store,
               itd.sku,
               itd.tran_date,
               SUM(itd.units),
               itd.sales_type
          FROM if_tran_data itd,
               win_skus wsk,
               win_store wst,
               domain_subclass ds
         WHERE wsk.forecast_ind = 'Y'
           AND itd.tran_code    = 1   /* sales tran_code
*/
           AND itd.store        = wst.store
           AND itd.sku          = wsk.sku
           AND itd.sku          = wst.sku
           AND ((itd.tran_date >
NVL(wst.last_sales_export_date, itd.tran_date - 1)
                    AND ds.load_sales_ind = 'N')
                OR (ds.load_sales_ind = 'Y'))
           AND wsk.dept         = ds.dept
           AND wsk.class        = ds.class
           AND wsk.subclass     = ds.subclass
           AND ds.domain_id     = :pi_restart_thread_val
           AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                 (itd.sku  =  :ps_restart_sku AND
                  itd.store > :ps_restart_location))
         GROUP BY itd.store,
                  itd.sku,
                  itd.tran_date,
                  itd.sales_type

        UNION ALL
```

```
SELECT itd.store,
       itd.sku,
       itd.tran_date,
       SUM(itd.units),
       itd.sales_type
  FROM if_tran_data itd,
       rag_skus_st rss,
       rag_style rs,
       domain_subclass ds
 WHERE rs.forecast_ind = 'Y'
   AND itd.tran_code   = 1
   AND itd.store       = rss.store
   AND rss.style       = rs.style
   AND itd.sku         = rss.sku
   AND ((itd.tran_date >
NVL(rss.last_sales_export_date, itd.tran_date - 1)
                 AND ds.load_sales_ind = 'N')
             OR (ds.load_sales_ind = 'Y'))
   AND rs.dept         = ds.dept
   AND rs.class        = ds.class
   AND rs.subclass     = ds.subclass
   AND ds.domain_id    = :pi_restart_thread_val
   AND (itd.sku > NVL(:ps_restart_sku,-999) OR
         (itd.sku   = :ps_restart_sku AND
          itd.store > :ps_restart_location))
 GROUP BY itd.store,
          itd.sku,
          itd.tran_date,
          itd.sales_type
ORDER BY 2, 1; /* sku, store */
```

Warehouse Issues Driving Cursor:

If the domain_type = ‘D’, the driving cursor is as follows:

```
        SELECT itd.wh,
               itd.sku,
               itd.tran_date,
               SUM(itd.units)
          FROM if_tran_data itd,
               win_skus wsk,
               win_wh wwh,
               domain_dept dd
         WHERE wsk.forecast_ind = 'Y'
           AND itd.tran_code    = 32   /* issues tran code
*/
           AND itd.wh           = wwh.wh
           AND itd.sku          = wsk.sku
           AND itd.sku          = wwh.sku
           AND ((itd.tran_date >
NVL(wwh.last_issues_export_date, itd.tran_date - 1)
                    AND dd.load_sales_ind = 'N')
                OR (dd.load_sales_ind = 'Y'))
           AND wsk.dept         = dd.dept
           AND dd.domain_id     = :pi_restart_thread_val
           AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                 (itd.sku   = :ps_restart_sku AND
                  itd.wh    > :ps_restart_location))
         GROUP BY itd.wh,
                  itd.sku,
                  itd.tran_date

        UNION ALL

        SELECT itd.wh,
               itd.sku,
               itd.tran_date,
               SUM(itd.units)
          FROM if_tran_data itd,
               rag_skus_wh rsw,
               rag_style rs,
```

```
                domain_dept dd
          WHERE rs.forecast_ind = 'Y'
            AND itd.tran_code   = 32 /* issues tran code */
            AND itd.wh          = rsw.wh
            AND rsw.style       = rs.style
            AND itd.sku         = rsw.sku
            AND ((itd.tran_date >
NVL(rsw.last_issues_export_date, itd.tran_date - 1)
                     AND dd.load_sales_ind = 'N')
                 OR (dd.load_sales_ind = 'Y'))
            AND rs.dept         = dd.dept
            AND dd.domain_id    = :pi_restart_thread_val
            AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                  (itd.sku   = :ps_restart_sku AND
                   itd.wh > :ps_restart_location))
          GROUP BY itd.wh,
                   itd.sku,
                   itd.tran_date
         ORDER BY 2, 1;  /* sku, wh */
```

If the domain_type = 'C', the driving cursor is as follows:

```
         SELECT itd.wh,
                itd.sku,
                itd.tran_date,
                SUM(itd.units)
           FROM if_tran_data itd,
                win_skus wsk,
                win_wh wwh,
                domain_class dc
          WHERE wsk.forecast_ind = 'Y'
            AND itd.tran_code    = 32   /* issues tran code
*/
            AND itd.wh           = wwh.wh
            AND itd.sku          = wsk.sku
            AND itd.sku          = wwh.sku
            AND ((itd.tran_date >
NVL(wwh.last_issues_export_date, itd.tran_date - 1)
                     AND dc.load_sales_ind = 'N')
                  OR (dc.load_sales_ind = 'Y'))
```

```
          AND wsk.dept         = dc.dept
          AND wsk.class        = dc.class
          AND dc.domain_id     = :pi_restart_thread_val
          AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                (itd.sku   = :ps_restart_sku AND
                 itd.wh    > :ps_restart_location))
        GROUP BY itd.wh,
                 itd.sku,
                 itd.tran_date

       UNION ALL

       SELECT itd.wh,
              itd.sku,
              itd.tran_date,
              SUM(itd.units)
         FROM if_tran_data itd,
              rag_skus_wh rsw,
              rag_style rs,
              domain_class dc
        WHERE rs.forecast_ind = 'Y'
          AND itd.tran_code   = 32 /* issues tran code */
          AND itd.wh          = rsw.wh
          AND rsw.style       = rs.style
          AND itd.sku         = rsw.sku
          AND ((itd.tran_date >
NVL(rsw.last_issues_export_date, itd.tran_date - 1)
                    AND dc.load_sales_ind = 'N')
                OR (dc.load_sales_ind = 'Y'))
          AND rs.dept         = dc.dept
          AND rs.class        = dc.class
          AND dc.domain_id    = :pi_restart_thread_val
          AND (itd.sku > NVL(:ps_restart_sku,-999) OR
                (itd.sku   = :ps_restart_sku AND
                 itd.wh > :ps_restart_location))
        GROUP BY itd.wh,
                 itd.sku,
                 itd.tran_date
```

```
ORDER BY 2, 1; /* sku, wh */
```

If the domain_type = ‘S’, the driving cursor is as follows:

```
SELECT itd.wh,
       itd.sku,
       itd.tran_date,
       SUM(itd.units)
  FROM if_tran_data itd,
       win_skus wsk,
       win_wh wwh,
       domain_subclass ds
 WHERE wsk.forecast_ind = 'Y'
   AND itd.tran_code    = 32   /* issues tran code
*/
   AND itd.wh           = wwh.wh
   AND itd.sku          = wsk.sku
   AND itd.sku          = wwh.sku
   AND ((itd.tran_date >
NVL(wwh.last_issues_export_date, itd.tran_date - 1)
                 AND ds.load_sales_ind = 'N')
        OR (ds.load_sales_ind = 'Y'))
   AND wsk.dept         = ds.dept
   AND wsk.class        = ds.class
   AND wsk.subclass     = ds.subclass
   AND ds.domain_id     = :pi_restart_thread_val
   AND (itd.sku > NVL(:ps_restart_sku,-999) OR
         (itd.sku   = :ps_restart_sku AND
          itd.wh    > :ps_restart_location))
 GROUP BY itd.wh,
          itd.sku,
          itd.tran_date


UNION ALL
```

```
SELECT itd.wh,
       itd.sku,
       itd.tran_date,
       SUM(itd.units)
  FROM if_tran_data itd,
       rag_skus_wh rsw,
       rag_style rs,
       domain_subclass ds
 WHERE rs.forecast_ind = 'Y'
   AND itd.tran_code   = 32 /* issues tran code */
   AND itd.wh          = rsw.wh
   AND rsw.style       = rs.style
   AND itd.sku         = rsw.sku
   AND ((itd.tran_date >
NVL(rsw.last_issues_export_date, itd.tran_date - 1)
                AND ds.load_sales_ind = 'N')
             OR (ds.load_sales_ind = 'Y'))
   AND rs.dept         = ds.dept
   AND rs.class        = ds.class
   AND rs.subclass     = ds.subclass
   AND ds.domain_id    = :pi_restart_thread_val
   AND (itd.sku > NVL(:ps_restart_sku,-999) OR
         (itd.sku   = :ps_restart_sku AND
          itd.wh > :ps_restart_location))
 GROUP BY itd.wh,
          itd.sku,
          itd.tran_date
ORDER BY 2, 1; /* sku, wh */
```

Output Specifications

Output Files

The four output files will be named as follows:

- Regular sales: rdsal*nn*.dat
- Promotion sales: pdsal*nn*.dat
- Clearance sales: cdsal*nn*.dat
- Warehouse issues: dlyiss*nn*.dat

Where *nn* is a two-digit number corresponding to the domain_id from which the data was derived, e.g. the regular sales output file from data in domain 13 would be named rdsal13.dat, the warehouse issue output file from data in domain 8 would be named dlyiss08.dat, and so on.

Output File Format:

Record Name	Field Name	Field Type	Default Value	Description
	Location	Char(20)		Store or Warehouse ID. Left-justified.
	SKU	Char(20)		SKU. Left-justified.
	Date	Char(8)		Transaction date. Left-justified. ('YYYYMMDD')
	Quantity	Char(13)		Sales/Issues. Contains up to four decimal places. If value is fractional, the quantity will be printed with decimal character '.' included, e.g. "1023.25". Right-justified.

Function Level Description

main():

The standard Retek main() function. Calls init(), process(), and final().

init():

Initialize restart recovery by calling retek_init() and set up the output files. The files should be named as described in Output Specifications. init() will also call a cursor to get the vdate from the period table and the domain_level from system_options.

format_buffer():

Formats the string which will be used to write to the output file.

init_arrays():

Allocates the necessary memory (sized to the restart_max_counter) to the arrays used in processing.

process():

This function makes a call to format_buffer() and init_arrays(). A restart flag indicating whether or not the sales loop was completed should be checked; if the loop did not complete (or this is the program's initial run), process_sales() is called, after which changes are committed and process_issues() is called, once complete, changes are committed. If the restart flag indicates that sales processing was complete at restart, skip process_sales() and call process_issues(), committing after completion.

process_sales():

This function contains the three sales driving cursors. Use the domain_level value fetched in init() to determine which cursor to use.

Open the appropriate driving cursor. In a while loop, perform an array fetch. If the cursor is exhausted, set a flag to indicate this. Use a for loop to loop through each of the records in the array. For each, call write_daily_sales(), passing the record's sales_type, store, SKU, tran_date and units as parameters. After each set of date is processed by the for loop, make a call to retek_force_commit(). If the exhausted cursor flag is set, break out of the while loop.

process_issues():

This function contains the three issues driving cursors. Use the domain_level value fetched in init() to determine which cursor to use. Open the appropriate driving cursor. In a while loop, perform an array fetch. If the cursor is exhausted, set a flag to indicate this. Use a for loop to loop through each of the records in the array. For each, call write_daily_issues(). After each set of date is processed by the for loop, make a call to retek_force_commit(). If the exhausted cursor flag is set, break out of the while loop.

write_daily_sales():

This function takes five strings as parameters: sales_type, store, SKU, tran_date, and units. If the sales_type is equal to "R" (regular sale), then write the data to the rdsal*nn*.dat file. If equal to "P" (promotion sale), write to the pdsal*nn*.dat file. If equal to "C" (clearance sale), write to the cdsal*nn*.dat file. All output will be written to files using the string formatted in format_buffer().

write_daily_issues():

This function takes four strings as parameters: wh, SKU, tran_date, and units. Write this data to the dlyiss*nn*.dat file using the string formatted in format_buffer().

final():

Take care of file clean up, freeing of memory, and complete the restart recovery process by calling retek_close().

Scheduling Considerations

This program should run daily in Phase 4 and, when run, should take the place of fdaydnld.pc unless a full download of forecast information is required. However, if a day's run is missed, then fdaydnld.pc must be run. Once fdaydnld.pc is run, then this program can be run the next day.

This program should run before the weekly sales/issues download programs: fisdnlds.pc, fisdnldf.pc, fsadnldf.pc, or fsadnlds.pc.

Restart/Recovery

This program will use restart recovery. The logical unit of work is each unique SKU/Location combination. The program will also maintain a restart flag which indicates whether or not the first process loop (sales) was complete at the time of restart. Rather than being included in a driving cursor's WHERE clause (as with most restart variables), this flag will only be used to determine which process loop to begin in on restart.