Retail passion. Innovative solutions.

Retek® Merchandising System™ 10.1.11

Operations Guide Addendum

Corporate Headquarters:

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

888.61.RETEK (toll free US)
Switchboard:
+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

Customer Support

Customer Support hours

Customer Support is available 7x24x365 via email, 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
Hong Kong	800 96 4262
Korea	00 308 13 1342
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

Sales History Rollup by Department, Class, and Subclass [hstbld]

Design Overview

The sales history rollup routine will extract sales history information for each item from the item_master, and item_loc_hist tables. The history information will be rolled up to the subclass, class, and dept level to be written to: dept_sales_hist, class_sales_hist, and subclass_sales_hist.

For each item, data to be saved includes sales qty, value, gross profit, and sales rate. This data must be collected from several tables including item_master, item_loc_hist and mask_rebuild. Letting the database (server) roll up the totals verse using a loop on the client enhances speed. Using a VIEW that contains all item information for the current week enhances simplicity. Data can then be summed from this single view instead of having to join across all 3 tables in a single select statement.

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 mask_rebuild table, which will determine what is rebuilt. This process is used after items are reclassified from one merchandise hierarchy to another. Rebuilding a department will rebuild each class and subclass within the department, thus, only one row is required on mask rebuild for the department. This type of rebuilding process will rebuild data from all dates on the item_master, item_loc_hist table, rolling them to the department, class, and subclass level.

Second, 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.

mask_rebuild table:

DEPT	CLASS	SUBCLASS	
X	NULL	NULL	Rebuild Department
X	X	NULL	Rebuild Class
X	X	X	Rebuild Subclass

TABLE	INDEX	SELECT	INSERT	UPDATE	DELETE
DEPT_SALES_HIST	No	Yes	Yes	Yes	No
CLASS_SALES_HIST	No	Yes	Yes	Yes	No
SUBCLASS_SALES_ HIST	No	Yes	Yes	Yes	No
ITEM_MASTER	No	Yes	No	No	No
ITEM_LOC_HIST	No	Yes	No	No	No
MASK_REBUILD	No	Yes	No	No	No
PERIOD	No	Yes	No	No	No
SYSTEM_VARIABLES	No	Yes	No	No	No

Scheduling Constraints

Processing Cycle:	PHASE 3 (weekly) PHASE AD-HOC (weekly)
Scheduling Diagram:	Must run after complete weekly sales have been updated by posupld. Also should be re-run on demand when a sales rollup request has been given for a given dept, class or subclass
Pre-Processing:	N/A
Post-Processing:	hstbld_post() Truncates the mask rebuild table.
Threading Scheme:	DEPT

Restart Recovery:

If program is run for reclassifying Items (first run time parameter = ‘rebuild’), the driving cursor for the program will be:

```
EXEC SQL DECLARE c_rebuild_eow CURSOR FOR
   SELECT im.dept,
          im.class,
          im.subclass,
          ilh.loc,
          to_char(ilh.eow_date,'YYYYMMDD'),
          ilh.week_454,
          ilh.month_454,
          ilh.year_454,
          ilh.sales_type,
          NVL(SUM(NVL(ilh.sales_issues,0)),0),
          NVL(SUM(NVL(ilh.value,0)),0),
```

```
          NVL(SUM(NVL(ilh.gp,0)),0)
     FROM item_master im,
          item_loc_hist ilh,
          v_restart_store rs
    WHERE im.item = ilh.item
      AND ilh.sales_type in ('P','R')
      AND ilh.eow_date = to_date(:ps_vdate,'YYYYMMDD')
      AND ilh.loc_type = 'S'
      AND rs.driver_value = ilh.loc
      AND rs.num_threads  = TO_NUMBER(:ps_num_threads)
      AND rs.thread_val   = TO_NUMBER(:ps_thread_val)
      AND (ilh.loc >= NVL(:ps_restart_store, -999) AND
                 (im.dept >= NVL(:ps_restart_dept,-999) AND
                     NVL(im.class,0) > NVL(:ps_restart_class,-999)
                  )
                )
GROUP BY im.dept,
          im.class,
          im.subclass,
          ilh.loc,
          to_char(ilh.eow_date,'YYYYMMDD'),
          ilh.week_454,
          ilh.month_454,
          ilh.year_454,
          ilh.sales_type
ORDER BY ilh.loc,
          im.dept,
          im.class,
          im.subclass,
          ilh.sales_type;
```

If program is run for current end of week (second run time parameter = ‘weekly’), the driving cursor for the program will be:

```
EXEC SQL DECLARE c_rebuild_dept CURSOR FOR
   SELECT im.dept,
          im.class,
          im.subclass,
          ilh.loc,
          to_char(ilh.eow_date,'YYYYMMDD'),
          ilh.week_454,
          ilh.month_454,
          ilh.year_454,
          ilh.sales_type,
          NVL(SUM(NVL(ilh.sales_issues,0)),0),
          NVL(SUM(NVL(ilh.value,0)),0),
          NVL(SUM(NVL(ilh.gp,0)),0)
     FROM item_master  im,
          item_loc_hist  ilh,
          hist_rebuild_mask hrm,
          v_restart_store rs
    WHERE im.item = ilh.item
      AND im.dept = hrm.dept
      AND ilh.sales_type in ('P', 'R')
      AND ilh.loc_type = 'S'
      AND rs.driver_value = ilh.loc
      AND rs.num_threads  = TO_NUMBER(:ps_num_threads)
      AND rs.thread_val   = TO_NUMBER(:ps_thread_val)
      AND (hrm.class IS NULL
           OR (im.class = hrm.class
               AND (hrm.subclass IS NULL
                    OR  im.subclass = hrm.subclass)))
      AND (ilh.loc >= NVL(:ps_restart_store, -999) AND
                    (im.dept >= NVL(:ps_restart_dept,-999) AND
                          NVL(im.class,0) > NVL(:ps_restart_class,-
999)
                     )
                )
```

```
GROUP BY im.dept,
    im.class,
    im.subclass,
    ilh.loc,
    to_char(ilh.eow_date,'YYYYMMDD'),
    ilh.week_454,
    ilh.month_454,
    ilh.year_454,
    ilh.sales_type
ORDER BY ilh.loc,
         im.dept,
         im.class,
         im.subclass,
         to_char(ilh.eow_date, 'YYYYMMDD'),
         ilh.sales_type;
```

Program Flow

N/A

Shared Modules

N/A

Function Level Description

init()

- Initialize restart recovery
- If processing current end of week, call check_eow_date()
- Initialize structure arrays for subclass_sales_hist insert, subclass_sales_hist update, class_sales_hist insert, class_sales_hist update, and sales_history fetch.
- Check_eow_date()
- Check that vdate is a valid end of week date

process()

- Open driving cursor
- Array fetch cursor
- Loop through array to process records.
- Increment dept and class sales variables for running totals of sales, value, gp, and forecast_sales.
- Call process_subclass().
- If dept/class/store/eow_date/sales_type changes, call process_class().
- If dept/store/eow_date/sales_type changes, call process_dept().
- Call process_subclass(), process_class(), and process_subclass() to process last record fetched (last record is not processed within above loop.
- Call insert_class()
- Call update_class()
- Call insert_subclass()
- Call update_subclass

Process_subclass()

- Fetch previous sales, value, gp, and forecast_sales value from subclass_sales_hist
- If fetch is not found (no record exists), add values to subclass_insert array
- If fetch is found,
  - Add values to subclass_update array
  - Increment delta variables to hold running totals for difference in subclass' sales and updated sales, value and updated value, gp and updated gp, forecast_sales and updated forecast_sales.

Process_class()

- Fetch rowid from class_sales_hist
- If fetch is not found, add values to class_insert array
- If fetch is found,
- Add values to class_update array
- Increment delta variables to hold running totals for difference in class' sales, value, gp, and forecast_sales by adding deltas from subclass' sales, value, gp, and forecast_sales.
- Reset subclass delta variables and class running total variables.

Process_dept()

- Perform update of dept_sales_hist
- If update is not found (record does not exist), perform insert into dept_sales_hist
- Reset class delta variables and dept running total variables.

Insert_class()

- Perform array insert of class_sales_hist.

Update_class()

- Perform array update of class_sales_hist.

Insert_subclass()

- Perform array insert of subclass_sales_hist.

Update_subclass()

- Perform array update of subclass_sales_hist.

I/O Specification

N/A

Technical Issues

N/A

Upload Customs tariff files [htsupld]

Design Overview

This batch program will be run whenever an updated US customs tariff file is available (probably twice a year) to upload HTS tariff information from the file into RMS HTS tables. The program will handle both the initial HTS information load as well as mid-year HTS updates that are supplied by the US government. The initial upload is handled by inserting information from the file into the tables; updating information already in the tables is handled by adjusting the effective dates of the existing HTS records and inserting a new set of HTS records into the tables.

Updating HTS records should follow the following guidelines:

- No HTS records with the same HTS and import country should have overlapping effect_from and effect_to dates. Import country is passed as an input parameter to the program, so that the program can support different import countries.
- The new HTS effective dates will never chop up the effective dates of an existing HTS, and there will never be any rollback in dates. Therefore, a new HTS can only start in the middle of an existing HTS or cover a completely different time frame after the existing HTS.
- When loading a new HTS that starts in the middle of an existing HTS, the effect_to date of the existing HTS should be adjusted to one day before the new effect_from date.
- No existing HTS information should be purged by the program. It's the client's responsibility to handle that.

TABLE	SELECT	INSERT	UPDATE	DELETE
HTS	Yes	Yes	Yes	Yes
HTS_TAX	No	Yes	Yes	Yes
HTS_FEE	No	Yes	Yes	Yes
HTS_OGA	No	Yes	Yes	Yes
HTS_TARIFF_TREATMENT	Yes	Yes	Yes	Yes
HTS_TT_EXCLUSIONS	No	Yes	Yes	Yes
TARIFF_TREATMENT	Yes	No	No	No
COUNTRY_TARIFF_TREATMENT	Yes	No	No	No
HTS_CHAPTER	Yes	No	No	No
OGA	Yes	No	No	No
UOM_CLASS	Yes	No	No	No
CODE_DETAIL	Yes	No	No	No
QUOTA_CATEGORY	Yes	No	No	No
COUNTRY	Yes	No	No	No
HTS_CVD	No	No	Yes	No

TABLE	SELECT	INSERT	UPDATE	DELETE
HTS_AD	No	No	Yes	No
HTS_REFERENCE	No	No	Yes	No
ITEM_HTS	Yes	Yes	Yes	No
ITEM_HTS_ASSESS	No	No	Yes	No
ORDSKU_HTS	Yes	Yes	Yes	Yes
MOD_ORDER_ITEM_HTS	No	Yes	No	No
PERIOD	Yes	No	No	No
SYSTEM_OPTIONS	Yes	No	No	No
DUAL	Yes	No	No	No
ORDSKU_HTS_ASSESS	No	No	No	Yes
ORDHEAD	Yes	No	No	No
ORDLOC	Yes	No	No	No
ORDSKU	Yes	No	No	No
CE_CHARGES	Yes	No	No	Yes
CE_ORD_ITEM	Yes	No	No	No
ITEM_SUPP_COUNTRY	Yes	No	No	No

Scheduling Constraints

Processing Cycle: Ad hoc

Scheduling Diagram: Run anytime as needed.

Pre-Processing: after hts upload conversion (ushts2rms – PERL script).

Post-Processing: None

Threading Scheme: None

Restart Recovery

This program supports Retek standard intermittent commit and file upload restart/recovery. Recommended commit counter is 2000 (commit after every 2000 tariff records are read). Input file names must end in a ".1" for the restart mechanism to properly parse the file name. Since 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. The user can fix the rejected records and process the reject file again.

Program Flow

N/A

Shared Modules

ITEM_HTS_SQL.DELETE_ASSESS – given the item, hts, import_country_id, origin_country_id, effect_to and effect_from, this function deletes the corresponding record from item_hts_assess.

ITEM_HTS_SQL.DEFAULT_CALC_ASSESS – given the item, hts, import_country_id, origin_country_id, effect_to and effect_from, this function inserts into item_hts_assess, it also will potentially call other package functions and update other tables.

LC_SQL.DELETE_LCORDAPP – given the order_no, this function deletes from lc_ordapply table.

OTB_SQL.ORD_UNAPPROVE – given the order_no, this function updates the otb table.

ITEM_ATTRIB_SQL.GET_STANDARD_UOM – given the item_no, item_type and indicator, this function returns the standard_uom, standard_class, and conv_factor.

UOM_SQL.CONVERT – given the to_uom, from_value, from_uom, item, supplier and origin_country, this function returns the to_value.

SQL_LIB.BATCH_MSG – returns error message information.

ORDER_HTS_SQL.DELETE_ASSESS -- given the order_no and seq_no, this function deletes from the ordsku_hts_assess table.

ORDER_HTS_SQL.DEFAULT_CALC_ASSESS -- given the order_no, seq_no, pack or item, hts, import_country_id, origin_country_id, effect_to and effect_from, this function inserts into ordsku_hts_assess, it also will potentially call other package functions and update other tables.

CE_CHARGES_SQL.INSERT_COMPS – given the ce_id, vessel_id, voyage_flt_ind, order_no, item, pack_item, hts, import_country_id, effect_from, effect_to, cvb_code this function inserts into the ce_charges table.

Function Level Description

Main

- Standard Retek main function. This program takes in four parameters: userid/passwd, input file, reject file, import country id.

Init

- A global variable is used to hold the import country id that is passed in as a program input parameter. Call check_country to make sure that import country exists on the COUNTRY table; return with fatal error if not. It is used as the import country throughout the program.
- Open input file for read and open reject file for write.
- Call retek_init() for restart/recovery initialization.
- If it is a fresh start, call retek_get_record to read the FHEAD line into the fhead structure.
- Fetch vdate from period table.
- Fetch max_item from hts table and max ct from ordsku_hts and max ct from ce_charges.
- Fetch update_item_hts_ind and update_order_hts_ind from the system_options table

- Call check_spi to make sure that ‘C1’ and ‘C2’ exist in the TARIFF_TREATMENT table as SPI’s. ‘C1’ and ‘C2’ are default tariff treatments for every HTS. Return with fatal error if not.

File_process

- Call function retek_get_record in a while loop to read the THEAD line into the thead structure:
  - if the record type returned is ‘FTAIL’, exit the loop;
  - set a save point.
  - If the record type returned is ‘THEAD’, read the THEAD line into the thead structure that contains V1, V2, V3, V4 fields. The V4 record is not currently used in RMS/RTM.
  - If the record type returned is other than ‘FTAIL’ or ‘THEAD’, give a fatal error (wrong record type).
- Call function process_THEAD to further process data contained in the THEAD. Set process error flag to indicate non-fatal process error.
- Call function retek_get_record in a while loop to read the TDETL line into the tdetl structure:
  - if the record type returned is ‘TTAIL’, exit to the outer loop to continue reading THEAD records if any exists;
  - if the record type returned is other than ‘TDETL’ or ‘TTAIL’, give a fatal error (wrong record type).
  - Call function process_TDETL to further process data contained in the TDETL. Set process error flag to indicate non-fatal process error.
- If update_item_hts_ind = “Y”,
  - If tran_code is “A” or “R”, call item_hts_update function. “A” stands for Update only and “R” stands for Replace. In both of these cases (as opposed to the other possibility of “D” for Delete) item tables will need to be updated.
- If update_order_hts_ind = “Y”, call ordsku_hts_search function.
- If process error flag is set. Rollback database process to the save point. Write rejected records to the reject file.
- Call restart_force_commit to perform intermittent commit for restart/recovery.

Process_THEAD

Fill the hts_keys structure with data from THEAD.

After fill in the hts_keys, verify that effect_from < effect_to date. If not, reject the record right away. Call valid_all_numeric function to check effect_from, and effect_to field. If invalid reject the record.

This function processes the information in V1, V2 and V3 records based on the transaction code ("A","R", "D") in the V1 record. It compares the new effective dates against those of any existing HTS records with the same HTS code and import country.

If the transaction code type is ‘A’, insert a record into the HTS table; if the transaction code type is ‘R’, update the HTS record that has the same HTS code, import country id, effect_from and effect_to dates

For transaction code "A":

If new HTS covers a time period different than and after any existing HTS, or no HTS exists for the given HTS/import country, is a valid record for inserting.

If new the HTS record is overlapping with existing record and its effect_from date > existing record and effect_to >= existing effect_to date, it is a valid record. Process is as follows:

1 Insert an HTS record with the same data as the existing overlapping HTS, except that the effect_to date should be 1 day before the effect_from date of the new HTS record;

2 Update the effect_to date of all corresponding child records to 1 day before the effect_from date of the new HTS record.

3 Insert new hts to the related tables.

Detailed technical description:

- Call function validate_hts_update to verify that the record is valid for insert/update to the database or reject to the reject file. For the valid record call hts_child_update function to prepare child table processing.
- Call hts_table_insert function to insert record to the hts table. if any invalid information exists, write to error file.
- Call hts_oga_insert function to insert record/s to the hts_oga table. if any invalid information exists, write to error file.
- Call hts_spi_insert function to insert record/s to the hts_tariff_treatment table. if any invalid information exists, write to error file.
- Call hts_gsp_insert function to insert record/s to the hts_tt_exclusions table. if any invalid information exists, write to error message log file.

Set process error flag if non fatal error occurs. Return error flag.

For transaction code "R":

1 Search for the HTS with the same HTS, import country id, effect_from and effect_to dates. If no record found, reject the record.

2 If a record is found, delete the following child table records with the same HTS, import country id, effect_from and effect_to dates:

3 Insert to update the HTS table and re-insert child table information from the input file.

Detailed Technical Description:

- Call function search_hts_update to find record that can be updated in the database tables.
- If one exists, prepare child tables for processing.
- Call hts_table_insert function to insert record to the hts table. if any invalid information exists, write to error file.
- Call hts_oga_insert function to insert record/s to the hts_oga table. if any invalid information exists, write to error file.
- Call hts_spi_insert function to insert record/s to the hts_tariff_treatment table. if any invalid information exists, write to error file.

- Call hts_gsp_insert function to insert record/s to the hts_tt_exclusions table. if any invalid information exist, write to error message log file.

Set process error flag if non fatal error occurs. Return error flag.

For transaction code "D":

1 Search for the HTS with same HTS, import country id , effect_from and effect_to dates.

2 If a record is found update HTS and all its child records to yesterday.

Note: Since the dates are still presented in 2-digit year in the 99 tape, we assume that the year coming in as 00-49 means 2000-2049, and 50-99 means 1950-1999. The customs uses '999999' to mean Dec 31st, 2039.

Detailed Technical Description:

Call function search_hts_reset to find updateable record in the hts table. If one exists, insert new hts record. Call function hts_child_update to update all the child records, then delete the existing hts record.

Validate_hts_update

1 new HTS starts before or on the same day as any existing HTS, or

new HTS starts after and ends before any existing HTS:
effect_from >= new effect_from OR
effect_from < new effect_from and effect_to > new effect_to

This is an invalid record. Write the record to the reject file, write an error message to the message log file, and return to the calling function with a non-fatal error.

2 new HTS starts after and overlaps with an existing HTS:

effect_from < new effect_from and effect_to >= new effect_from or new HTS starts after old end date and therefore does not overlap at all. The ranges are completely separate.

This is a valid record, and a most likely scenario. Fetch the effect_from and effect_to of the existing HTS. Insert a new record with effect_from date same as existing overlapping hts record and effect_to date is 1 day before the new effect_from date to hts table.

Call function hts_child_update function to update effect_to date of all child records to 1 day before the new effect_from date.

Delete the old record from hts table.

Search_hts_update

1 Search for the HTS with the same HTS, import country id, effect_from and effect_to dates. If no record found, reject the record.

2 If a record is found, delete the following child table records with the same HTS, import country id, effect_from and effect_to dates:

HTS_TT_EXCLUSIONS

HTS_TARIFF_TREATMENT

HTS_OGA

HTS_TAX

HTS_FEE

Note: HTS table record cannot be deleted due to the other child tables on HTS: ITEM_HTS, ITEM_HTS_ASSESS, ORDSKU_HTS, HTS_CVD, HTS_AD, HTS_REFERENCE, HTS_CHAPTER. The information on these tables won't be loaded in the HTS upload process.

Seach_hts_reset

1 Search for the HTS with the same HTS, import country id, effect_from and effect_to dates. If no record found, reject the record.

2 Insert into HTS, all the same information, but inserting yesterday as the new to_date.

3 If a record is found, call hts_child_update function to update the records in the child tables with effect_to date to yesterday:

Hts_child_update

This function updates the effect_to date of the existing overlapping HTS record on child tables. Since the child tables have referential constraints on the effective dates of the parent table HTS.

Update the effect_to date of all corresponding child records to 1 day before the effect_from date of the new HTS record.

The following child tables should be updated:

- HTS_TARIFF_TREATMENT
- HTS_TT_EXCLUSIONS
- HTS_AD
- HTS_CVD
- HTS_OGA
- HTS_REFERENCE
- HTS_TAX
- HTS_FEE
- ITEM_HTS
- ITEM_HTS_ASSESS
- ORDSKU_HTS
- CE_CHARGES

Note: Since table HTS_TT_EXCLUSIONS has a foreign key on the effect_to date of table HTS_TARIFF_TREATMENT, we cannot update the effect_to date of HTS_TARIFF_TREATMENT directly. Likewise, insert an HTS_TARIFF_TREATMENT record with the new effect_to date first; then update the effect_to date of the HTS_TT_EXCLUSIONS table; at the end delete the HTS_TARIFF_TREATMENT record with the original effect_to date.

Call delete_ord_temp_tables and pass in the value "-1" because there is no known order_no at this point.

Item_hts_update

1 Call size_item_array function to allocate space for the items

2 Fetch item, origin_country_id and status from item_hts into struct

3 If no data found, call free_itemlist and go to the next record. If data is found,

4 Loop

- If tran_code = “A” the item will need to be inserted with the same data as the fetched record but with new effect_to and effect_from dates.
  - Insert dates into item_hts
  - Delete old record from item_hts
  - call the package SQL Delete assess to delete the old records from item_hts_assess.
- If tran_code = “R”
  - Call SQL Delete_assess to delete the old records from item_hts_assess
- Call SQL Default_calc_assess to update the item_hts_assess table (ie insert record with new dates and recalculate)
- Call ECL_CALC_SQL.CALC_COMP to recalculate expenses based on new assesses.
- Insert into mod_order_item_hts a new record with same data but new dates.
- Call free_itemlist

Ordsku_hts_search

1 Call size_ord_array function to allocate space for the order information

2 Fetch values from ordhead, ordsku_hts, ordsku and ordloc into struct (all necessary values to be able to do a complete insert into the mod_order_item, ordsku_hts, and ordsku_hts_assess tables.

3 If no data found, call free_ordlist and go to next record. If data is found,

4 Loop

- If order status = “A”, (the order needs to be updated) set status from approved back to worksheet by calling SQL functions (LC_SQL.DELETE_LCORDAPP and OTB_SQL.ORD_UNAPPROVE).
- Insert into mod_order_item_hts table (just the order_no and indicator set to ‘Y’)
- Call ordsku_hts_update
- Call free_ordlist

Ordsku_hts_update

Call size_ce_array to allocate space for the custom entry information

- Fetch custom entry values from ce_ord_item, ce_head, item_supp_country into struct
- If no data found, call ordhts_update. If data is found,
  - If CE status = "W", (worksheet status)
    - Call ordhts_update
    - Loop for each custom entry record
      - Call ce_update
  - If status != "W" then the quantity cleared will need to be compared to the total quantity. In order to do that they will need to be converted to the standard uom format
    - Loop
      - Call uom_convert to get the total quantity.
      - If total_qty < qty_ordered
        - Call ordhts_update
- Call free_ceordlist

Ordhts_update

1 if tran_code = "A" or "D"
  - Delete old record (record with old dates) from ordsku_hts_assess
  - Delete old record (record with old dates) from ordsku_hts
  - Insert record with new dates into ordsku_hts

2 Else if tran_code = "A"
  - Insert record with new dates into ordsku_hts

3 else if tran_code = "D"
  - Call SQL Delete_assess by calling order_del_assess function
  - Call SQL calc_comp
  - If the item is a pack item check to see if a record already exists on mod_order_item_hts – if it does not, insert one with the pack_item
  - If it is not a pack item, insert with item_no into mod_order_item_hts.
  - Return 0

4 Else if tran_code = "R"
  - Call delete_ord_temp_tables and pass in the order_no.

5 Call SQL Delete_assess by calling order_del_assess function

6 Call ORDER_HTS_SQL.DEFAULT_CALC_ASSESS with either the pack_no or item_no depending on if it is a pack or not.

7 Call ELC_CALC.CALC_COMP

8 If it is a pack item insert into mod_order_hts with the pack_no

9 If it is not a pack item, insert into mod_order_item_hts with the item_no

Ce_update

1 Delete from ce_charges.

2 if it is a “D”, call CE_CHARGES_SQL.INSERT_COMPS

Hts_table_insert

Before inserting into or updating the HTS table,

1 Call function check_chapter to make sure that the chapter already exists on the HTS_CHAPTER table. If not, reject the record;

2 Call check valid_all_numeric function to check unit for all numeric value.

3 Call function check_uom to make sure that the UOMs (UOM1, UOM2, UOM3) already exist on the UOM_CLASS table. Reject the record if UOM does not exist.

4 Call function check_duty to make sure that the duty code already exists on the CODE_DETAIL table. If not, reject the record.

5 Call valid_all_numeric function to verify that the quota is all numeric. Then calling function check_quota to make sure that the quota category already exists on the QUOTA_CATEGORY table. If not, reject the record.

Update the existing hts record with the updated hts_desc, chapter, units, units_1, units_2, units_3, duty_comp_code, more_hts_ind, quota_cat, quota_ind, ad_ind, cvd_ind.

Insert the following into the HTS table:

- hts: tariff number (V1c)
- import_country_id: import country from the program input parameter
- effect_from: begin effective date (V1e)
- effect_to: end effective date (V1f)
- hts_desc: commodity description (V1l)
- chapter: 1st 4 (leftmost) digits of tariff number
- units: number of reporting units (V1g)
- units_1: first unit of measure (V1h) (If the number of reporting units is zero, this should be defaulted to ‘X’)
- units_2: second unit of measure (V1I) –NULL if not given
- units_3: third unit of measure (V1j)—NULL if not given
- duty_comp_code: duty code (V1k)
- more_hts:Y if additional tariff indicator (V2j is ‘R’, N otherwise
- quota_cat: category number (V3h) but only if quota indicator (V3g) is 1
- quota_ind ‘Y’ if there is a quota,’N’ otherwise
- ad_ind ‘Y’ if the anti-dumping flag (V3f) is 1, N otherwise

- cvd_ind ‘Y’if the countervailing duty flag (V2k) is 1, N otherwise

Hts_oga_insert

For each OGA code, call function check_oga to verify that the OGA code exists on the OGA table. If not, reject the record; otherwise, call hts_oga_insert to insert into HTS_OGA.

- Insert the following into the HTS_OGA table:
- hts: tariff number (V1c)
- import_country_id: import country from the program input parameter
- effect_from: begin effective date (V1e)
- effect_to: end effective date (V1f)
- code: OGA code from OGA codes field (V3f)
- reference_id: NULL
- comments: NULL

Hts_spi_insert

For each SPI, call function check_spi to check if the SPI exists on the tariff_treatment table; if not, reject the record. Call function hts_tariff_treatment_insert to insert into HTS_TARIFF_TREATMENT. In addition to the SPI records in V3, ‘C1’ and ‘C2’ are default tariff_treatments for every HTS. So, two extra records should be inserted into HTS_TARIFF_TREATMENT with SPI codes ‘C1’ and ‘C2’. ‘C1’ takes the special_duty_rate from V1 and Column 1 rates from V2; ‘C2’ takes Column 2 rates from V2.

Before inserting, call function check_spi to make sure that the SPI code (tariff treatment) exists on the TARIFF_TREATMENT table; reject the record if it does not.

Call valid_all_numeric function to check specific_rate, ad_rate, other_rate for all numeric value. If not, reject the record.

Reject HTS lines that have rate greater than 9999999999. A brief explanation of why this is done is located at the end of the function level description section.

Insert the following into the HTS_TARIFF_TREATMENT table:

- hts: tariff number (V1c)
- import_country_id: import country from the program input parameter
- effect_from: begin effective date (V1e)
- effect_to: end effective date (V1f)
- tariff_treatment: SPI code from V3i
- specific_rate: 0,col1 or col2 specific rate, as appropriate (0 for SPI’s,col 1 for col1, col 2 for col2)
- av_rate: 0,col1, or col2 ad valorem rate, as appropriate (0 for SPI’s)
- other_rate: 0,col1, or col2 other rate, as appropriate (0 for SPI’s)

Hts_gsp_insert

For each GSP excluded country, call function check_country_tariff_treatment to check that the country and tariff treatment combination exists on the COUNTRY_TARIFF_TREATMENT table; if not, reject the record.

Insert the following into the HTS_TT_EXCLUSIONS table

- hts: tariff number (V1c)
- import_country_id: import country from the program input parameter
- effect_from: begin effective date (V1e)
- effect_to: end effective date (V1f)
- tariff_treatment: first SPI code from V3i
- origin_country_id: excluded country code from V3d (GSP excluded countries)

Check_spi

Check to see if SPI exists on TARIFF_TREATMENT table; reject the record if it doesn't.

Check_country

Check to see if country exists on COUNTRY table; reject the record if it doesn't.

Check_chapter

Check to see if chapter exists on the HTS_CHAPTER table and reject the record if it doesn't.

Check_uom

Check to see if uom exists on UOM_CLASS table; reject the record if it doesn't.

Check_duty

Check to see if duty code exists on CODE_DETAIL table (check for the code where code_type='DCMP'); reject the record if it doesn't.

Check_quota

Check to see if the quota_category exists on the QUOTA_CATEGORY table; reject the record if it doesn't.

Check_oga

Check to see if the oga code exists on the OGA table; reject the record if it doesn't.

Check_comb_country_tt

Check to see if the country and tariff_treatment combination exists on the COUNTRY_TARIFF_TREATMENT table; reject the record if it doesn't.

Process_TDETL

Format the tax line information from tdetl structure.

Call function process_taxfee, if no non-fatal error in the process_THEAD function.

Process_tax_fee

If tax specific rate or tax ad rate is not null, call hts_taxfee_insert to insert the tax rates into HTS_TAX or HTS_FEE tables. If special rates exist on the tax line, call function hts_tariff_treatment_insert to insert into the HTS_TARIFF_TREATMENT table using the ISO country code as the tariff treatment (SPI). If the SPI given on the tax line already exists for the HTS, the record should be updated, as the tax line special rate takes precedence over the V3 line SPI's rate

Call valid_all_numeric function to check tax_specific_rate, tax_av_rate, fee_specific_rate, fee_av_rate for all numeric value, if not reject the record.

Reject HTS lines that have rate greater than 9999999999. A brief explanation of why this is done is located at the end of the function level description section.

Hts_taxfee_insert

If the tax class code is 016,017,018,or 022 it is a tax; insert into HTS_TAX

If the tax class code is 038,053,054,055,056,057,079,090,103 it is a fee; insert into HTS_FEE

Insert the following into the HTS_TAX or HTS_FEE table:

- hts: tariff number (V1c)
- import_country_id: import country from the program input parameter
- effect_from: begin effective date (V1e)
- effect_to: end effective date (V1f)
- tax_type/fee_type: tax class code (V5h)
- tax_comp_code/fee_comp_code: tax comp code (V5i)
- tax_specific_rate/fee_specific_rate: tax specific rate (V5k)
- tax_av_rate/fee_av_rate: tax ad valorem rate (V5l)

Hts_tariff_treatment_insert

Before calling this function, call function check_spi to make sure that the SPI code (tariff treatment) exists on the TARIFF_TREATMENT table; reject the record if it does not.

Insert the following into the HTS_TARIFF_TREATMENT table:

- hts: tariff number (V1c)
- import_country_id: import country from the program input parameter
- effect_from: begin effective date (V1e)
- effect_to: end effective date (V1f)
- tariff_treatment: SPI code from V3i and VDc
- specific_rate: 0,col1 or col2 specific rate, as appropriate (0 for SPI's,col 1 for col1, col 2 for col2)
- av_rate: 0,col1 or col2 ad valorem rate, as appropriate (0 for SPI's)
- other_rate: 0,col1 or col2 other rate, as appropriate (0 for SPI's)

Size_item_array

Allocates space for the item array struct

Size_ord_array

Allocates space for the order array struct

Size_ce_array

Allocates space for the custom entry array struct

Free_orditemlist

Frees the space in the array

Free_itemlist

Frees the space in the array

Free_ceordlist

Frees the space in the array

Uom_convert

Calls ITEM_ATTRIB_SQL.GET_STANDARD_UOM

Calls UOM_SQL.CONVERT

Order_del_assess

Calls ORDER_HTS_SQL.DELETE_ASSESS

Delete_ord_temp_tables

If an order no is not passed in, look at the hts table and see if there is an order that exists for that hts. If so, loop and for each record see if there is a record to delete on the temp tables by calling ORDER_SETUP_SQL.DELTE_TEMP_TABLES.

If the order number was passed in, call ORDER_SETU_SQL.DELETE_TEMP_TABLES right away.

Final

Restart/recovery close and close input and reject file.

Why HTS lines that have a rate greater than 9999999999 need to be rejected:

For fields specific_rate, av_rate, other_rate, RMS has the data type Number(12,8) and numbers coming in from the customs tape also have 8 implied digits. However, when storing the number into the Retek database, we need to divide the number coming in from the customs tape by 1000000 (left shift 6 digits) instead of 100000000 (left shift 8 digits). This is because Retek stores the percent part of the rate only. In other words, rate 11.5% (0.115) is stored as 11.5 in Retek database, whereas it will come in from the customs tape as 11500000 (=0.115). Therefore, the highest rate that can be represented in Retek is 9999.99999999% (= 99.9999999999, or < 100 times). So we need to reject HTS lines that have rate greater than 9999999999.

Note: This is true for hts spi and hts tax/fee specific_rate and av_rate, except that when 999999999999 (12 nines) are used, it represents a special code for NULL.

I/O Specification

Here is the layout of the input file to be uploaded:

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
	Retek file ID	Char(5)	HTSUP	Describes file type
THEAD	Record descriptor	Char(5)	THEAD	Describes file line type
	Line number	Number(10)		Sequential file line number
	Transaction id	Number(14)		Unique transaction id
	HTS Line	Char(358)		V1 through V4 records from the customs HTS file concatenated together
TDETL	Record descriptor	Char(5)	TDETL	Describes file line type
	Line number	Number(10)		Sequential file line number
	Transaction id	Number(10)		Unique transaction id
	Tax/fee line	Char(80)		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)		Sequential file line number
	Detail lines	Number(6)		Number of lines between THEAD and TTAIL
FTAIL	Record descriptor	Char(5)	FTAIL	Describes file line type
	Line number	Number(10)		Sequential file line number
	Transaction lines	Number(10)		Number of lines between FHEAD and FTAIL

Here is the layout of the original input file:

Note: The input file contains lines of 2400 characters, i.e. 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 RMS/RTM. 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.

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)		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	begin effective date	char(6)		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)		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.

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)		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)		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)		A code representing the third unit of measure. Valid unit of measure codes are listed in Appendix C.
k	duty computation code	char(1)		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)		A condensed version of the commodity description that appears in the HTS.
m	column 1 specific rate of duty	Number(12)		The rate of duty that appears in the General column of the HTS. Eight decimal places are implied.

Record Name	Field Name	Field Type	Default Value	Description
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 RMS.
o	space fill	char(1)	blank	Space fill. Not used in RMS.
V2 a	Control identifier	char(1)	V	Identifies start of record
b	Record type	char(1)	2	Identifies record type
c	tariff number	Number(10)		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)		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)		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	Number(12)		The specific rate of duty that appears in Column 2 of the HTS. Eight decimal places are implied.
g	Column 2 ad valorem percentage	Number(12)		The ad valorem rate of duty that appears in Column 2 of the HTS. Eight decimal places are implied.

Record Name	Field Name	Field Type	Default Value	Description
h	Column 2 other rate	Number(12)		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)		A code indicating if a tariff number may be subject to a miscellaneous permit/license number.
l	space fill	char(4)	blanks	Not used in RMS.
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)		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.

Record Name	Field Name	Field Type	Default Value	Description
d	GSP excluded countries	char(20)		The International Organization for Standardization (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.
e	OGA codes	char(15)		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)		A code located in the HTS indicating the textile category assigned to the tariff number. If there is no textile category number, space fill.

Record Name	Field Name	Field Type	Default Value	Description
I	special program indicators	char(28)		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 RMS.
b	Record type	char(1)	4	identifies record type
c	tariff number	number(10)		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)		A code representing the value edit.
e	value low bounds	number(10)		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)		A value representing the maximum value edit. Five decimal places are implied. If this record contains date edits (positions 36-53), space fill.

Record Name	Field Name	Field Type	Default Value	Description
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)		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.
I	end restriction date	char(4)		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.
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)		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)		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)		A code representing the value edit.

Record Name	Field Name	Field Type	Default Value	Description
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.
o	quantity edit code	char(3)		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)		A code representing the first entry date restriction code.
q	high quantity	number(10)		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)		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.

Record Name	Field Name	Field Type	Default Value	Description
d	Country code	char(2)		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)		The specific rate of duty listed in the Special column of the HTS. Eight decimal places are implied.
f	ad valorem rate	number(12)		The ad valorem rate of duty listed in the Special column of the HTS. Eight decimal places are implied.
g	Other rate	number(12)		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)		A code representing the tax/fee class. Valid tax/fee class codes are listed in Appendix B.
I	tax/fee comp code	char(1)		A code indicating the first tax/fee computation formula. Computation formulas are presented in Appendix F.

Record Name	Field Name	Field Type	Default Value	Description
j	tax/fee flag	number(1)		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 RMS.
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.
m	space fill	char(1)	blank	Space fill.
V6 through VC records have the same fields as the V5 record.				
NEWLINE			\n	
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)		unique tariff number
d	Special Program Indicator (SPI) Code	char(32)		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)		Space fill.

Store Add [storeadd]

Design Overview

This program will add all information necessary for a new store to function properly. When a store is added to the system, the store will be accessible in the system only after storeadd.pc is run.

The batch program loops through each record on the store_add table.

Also, it supports the replenishment system in RMS.

Scheduling Constraints

Processing Cycle:	Daily, Ad Hoc Phase
Scheduling Diagram:	N/A
Pre-Processing:	pcext.pc pcdnld.pc
Post-Processing:	likestore.pc slocrbld.pc
Threading Scheme:	Table based processing, do not use multithreading.

Restart/Recovery

Select ALL FIELDS from store_add.

After a record on store_add has been processed successfully, it is immediately deleted if like-store functionality is not used. If like-store functionality is used, other checks are executed. Thus, restart recovery is implicit in storeadd.pc.

Program Flow

N/A

Function Level Description

init()

Declare restart variables

Get system variables (ELC indicator and pricing rule)

process()

Loop through store_add table

Skip records that have already been processed by storeadd but not likestore.pc.

Set “new” variable indicators

If like-store functionality is used, set the store open and store close date to a date far in the future to prevent the new store from being used in the system until the all the store addition batch programs are completed. At that point, the store open and store close dates will be updated to the correct dates.

Insert into store table

Call Insert_Pricing_Zone

If elc_ind = ‘Y’

Call Insert_Cost_Zones
end if;
If copy_close_ind = ‘Y’
Call Copy_Close_Sched
End if;
If copy_dlvry_ind = ‘Y’
Call Copy_Dlvry_Sched
End if;
If copy_activity_ind = ‘Y’
Call Copy_close_sched
End if;
Call Insert_Stock_Loc_Traits
If (like_store_ind)
Delete from store_add
Call update_regionality_matrix
Call update_loc_sec_ind to allow user location security to be rebuilt.
If like_store_ind = 0
Call insert_pos_store
Call retek_force_commit after all processing is done for each store
Insert_Pricing_Zone()
This function inserts records into pricing zone tables as is appropriate to the store being created:
insert corporate pricing zone information
insert store pricing zone information
call Item_Zone_Price
if new_price_zone_ind = ‘N’
insert zone info for existing currency
else
insert new zone info
call Item_Zone_Price (to add appropriate record for the new zone)
Insert_Cost_Zones()
This function inserts records into cost zone table as is appropriate to the store being created:
insert corporate cost zone information
insert store cost zone information
if new_cost_zone_ind = ‘N’
insert cost zone detail records

else

insert new zone

Item_Zone_Price()

This function inserts records into the item_zone_price table for a new pricing zone after it's been created.

Copy_Close_Sched()

This function copies all the location closed information from the selected like_store which the close_date are greater or equal to current and copies them into location_closed and company_closed_excep tables for the new store.

Copy_Dlvry_Sched()

This function copies all the location delivery schedules from the selected like_store and copies them into the loc_dlvry_sched, loc_dlvery_sched_days, and loc_dlvry_sched_exc tables for the new store.

Insert_Stock_Loc_Traits()

This function calls the stkledgr_sql.stock_ledger_insert and loc_traits_sql.new_org_hier package functions, which insert records into the stock ledger and hierarchy tables.

update_regionality_matrix

This function calls regionality_matrix_sql.new_store to make sure the regionality matrix tables are maintained/inserted for the new store.

update_loc_sec_ind()

This function set update_loc_sec_ind in system_variables table to 'Y' if security policies exist in order to allow the user location security to be rebuilt.

insert_pos_store()

This function insert data into POS_STORE table based on a similar record.

final()

This function stops restart recovery.

I/O Specification

N/A

Technical Issues

N/A