

Batch Detail Design

I. Functional Area

Promotion

II. Module Affected

Prmpcupd.pc – Promotion Price Update

III. Design Overview

This new program will update item_loc table with promotion price information. It will update the promotion fields, promo_retail, promo_selling_retail and promo_selling_uom, in the item_loc table with promotion price information when a simple promotion applies the item/location combination. It will also update these promotion fields to null when a promotion ends today. In addition, it will update the item_loc table with any promotion changes, including add promotion items to the extracted promotions, delete promotion items from the extracted promotions and make promotion price changes to the extracted promotions. If the Batch with Online Users indicator is 'Y' it will bulk lock the item_loc records first before performing the necessary updates. Records that were processed successfully will be deleted from the prmpcupd_item_loc_temp temporary table, otherwise it will remain in the temporary table so that it can be processed in the next batch run. Information about the records that were not processed due to locking will be inserted to the batch_lock_log table.

This program will run daily in nightly batch cycle and should be run after the prmxext.pc.

| TABLE | INDEX | SELECT | INSERT | UPDATE | DELETE |
|------------------------|-------|--------|--------|--------|--------|
| promhead | No | Yes | No | No | No |
| promstore | No | Yes | No | No | No |
| Promsku | No | Yes | No | No | No |
| Item_loc | No | No | No | Yes | No |
| period | No | Yes | No | No | No |
| dual | No | Yes | No | No | No |
| Item_master | No | Yes | No | No | No |
| prmpcupd_item_loc_temp | No | Yes | No | No | Yes |
| system_options | No | Yes | No | No | No |
| item_loc_lock_temp | No | Yes | Yes | No | Yes |
| batch_lock_log | No | Yes | Yes | No | Yes |

IV. Stored Procedures / Shared Modules (Maintainability)

PROMOTION_ATTRIB_SQL.GET_PROMO_RETAIL – Returns the promotion retail price for the item/location.

UOM_SQL.CONVERT – converts the values between two UOMs.

V. Input Specifications



VI. Output Specifications

VII. Function Level Description

- Define a structure that will be used to define the driving cursor array.
- Define another structure to be used to define the update array.

Main():

This function should follow standard Retek main function format. It should call init(), process() and final() function.

Init():

This function performs preliminary processing and populates global variables. It will call retek_init function to handle the restart recovery logic and bring back the bookmark string in case of a restart. It will also retrieve the date of today (vdate) and tomorrow (vdate + 1) to be used in retrieving the valid promotions to process. It will retrieve the btch_w_usr_ind from the system_options table to determine users can be online at any time. If the Batch with Online Users indicator is set to 'Y', prmpcupd.pc's records in the batch_lock_log will be deleted if the batch is run for the first time.

Process():

This function will select the promotion, promotion store, store promotion start date, store promotion end date, promotion currency code, promotion item, standard unit of measure, promotion item status, promotion item price change type, amount, selling unit of measure, adjust type and the price ends in from the PROMSTORE, PROMHEAD, PROMSKU and ITEM_MASTER table. It will then call process_prom_item_loc to check the item/location relation and retrieve the promotion price. It will also call the update_item_loc function to update the item_loc table with the promotion price. The driving cursor should be similar as follows:

```
SELECT ph.promotion,
       ph.currency_code,
       TO_CHAR(ps.start_date, 'YYYYMMDD'),
       TO_CHAR(ps.end_date, 'YYYYMMDD'),
       ps.store,
       im.item,
       im.standard_uom,
       psku.status,
       psku.change_type,
       NVL(psku.change_amt,0),
       NVL(psku.selling_uom,''),
       psku.adjust_type,
       NVL(psku.ends_in,0)
FROM v_restart_store rv,
     promstore ps,
     promhead ph,
     promsku psku,
     Item_master im
WHERE ph.promotion = ps.promotion
      AND ph.status in ('E', 'M')           /* all extracted promotions */
      AND ps.extract_status in ('E', 'M')   /* all extracted promotion stores */
      AND psku.promotion = ps.promotion
      AND ps.start_date <= :tomorrow
      AND ps.end_date >= :today
      AND ps.promotion = psku.promotion
```



```

AND psku.change_type != 'EX'
AND im.item_level <= im.tran_level
AND (im.item = psku.item
      OR (im.item_parent = psku.item
          AND (psku.diff_id is null
                OR (psku.diff_id is not null
                    And (psku.diff_id = im.diff_1
                        OR psku.diff_id = im.diff_2
                        OR psku.diff_id = im.diff_3
                        OR psku.diff_id = im.diff_4))))))
OR(im.item_grandparent = psku.item
    AND(psku.diff_id is null
        OR(psku.diff_id is not null
            AND (psku.diff_id = im.diff_1
                OR psku.diff_id = im.diff_2
                OR psku.diff_id = im.diff_3
                OR psku.diff_id = im.diff_4))))))
AND rv.driver_value = ps.store
AND rv.driver_name = :ora_restart_driver_name
AND rv.num_threads = :ora_restart_num_threads
AND rv.thread_val = :ora_restart_thread_val
AND (ps.promotion > NVL(:ora_restart_promotion, -999) OR
      ((ps.promotion = :ora_restart_promotion) AND
        (ps.store >= :ora_restart_store)))
ORDER BY 1,5;

```

The flow of this function should be as follows:

- Define an array, la_prom_store, to hold the data fetched by the driving cursor.
- Define another array, la_item_loc, to hold the promotion data to be updated to the item_loc table.
- Call function size_prom_array().
- Call function size_item_loc_array().
- Open the driving cursor
- Array fetch the driving cursor to the la_prom_store array for commit_max_ctr records.
- Acquire lock to the item_loc records by inserting the item/location/rowid to the item_loc_lock_temp table and calling the function check_lock(). If there are records that are locked by another process, a flag will be set and information about the records that will not be processed will be written to the batch_lock_log table.
- In a for loop, loop through each record in the la_prom_store array
 - Call function process_prom_item_loc(). Pass in all the elements in current record to the function, as well as the item_loc array.
 - If the current promotion/store combination is different from last one, and the count of the item_loc array is greater than zero, call update_item_loc() to update the item_loc table. Pass in the item_loc array and the count of the records in the array. Note the count needs to be reset after the update.
- Commit and restart/recovery logic.
- Remember to update item_loc table with the last set of records in the item_loc array.

Size_prom_array():

This function will allocate memory for the array la_prom_store to size of commit_max_ctr.

Size_item_loc_array():

This function will allocate memory for the item_loc array to size of commit_max_ctr.

Process_prom_item_loc():

This function should process as follows:



- Define local variables to hold the promo_retail, promo_selling_retail, promo_selling_uom, and rowid. Initialize these variables to null.
- Create a cursor c_item_loc to retrieve the promo_retail, promo_selling_retail, promo_selling_uom and rowid from item_loc where status is 'Active' and the item, location match the passed in item and location.
- If no record found, return true. If error found, return fatal.
- If the end of store promotion date is today, call populate_item_loc_array function. Pass in the item_loc array and the local variables item, location, promo_retail, promo_selling_retail, promo_selling_uom and rowid. Return whatever is returned from the populate_item_loc function.
- Check the store Promotion start date:
 - If the start date is less than or equal to tomorrow, and the promsku.status is 'DI' (Deleted Item) or 'DP' (Delete Processed), check
 - if the promo_selling_unit_retail is null. If it is null, stop further processing and the function should return true.
 - if the promo_selling_unit_retail is not null, call populate_item_loc_array function. Pass in the item_loc array and the local variables item, location, promo_retail, promo_selling_retail, promo_selling_uom and rowid. Return whatever is returned from the populate_item_loc function.
 - Otherwise,
 - Call get_prom_retail() to retrieve the promotion price. Pass in the current records in the promo_store array and the local variable promo_selling_retail. Return false if the function call failed.
 - Compare the promo_selling_retail and the promo_selling_uom obtained in the get_prom_retail with the promo_selling_retail and promo_uom in the item_loc table (retrieved from cursor c_item_loc). If they are same, stop further processing and return true.
 - Call calc_std_retail() to convert the promotion retail to standard retail. Pass in the selling_uom, standard_uom, promo_selling_retail, promo_retail, item and location. Return false if the function call failed.
 - Call populate_item_loc_array(). Pass in the item_loc array and the local variables item, location, promo_retail, promo_selling_retail, promo_selling_uom and rowid. Return whatever is returned from the populate_item_loc function.

Populate_item_loc_array():

This function will process as follows:

- Check if the count of the total records in the item_loc array plus 1 is greater than the size of the item_loc array. If it exceeds the array size, call resize_item_loc_array().
- Copy the promo_retail, promo_selling_retail, promo_selling_uom and rowid to the item_loc array. Populate the item_loc array's last_update_datetime and last_update_id with the SYSDATE and UESER, respectively.
- Return true.

Resize_item_loc_array():

This function will allocate additional max_commit_ctr memory for the item_loc array.

Get_prom_retail():

This function will call stored procedure PROMOTION_ATTRIB_SQL.GET_PROMO_RETAIL to retrieve the promo_selling_retail.

Calc_std_retail():

This function will first retrieve the convert factor from a selling_uom to standard_uom. Then calculate the promotion retail per standard unit of measure. It should process as the steps described below:

- Call stored procedure UOM_SQL.CONVERT to find the convert factor. Pass 1 to the from_value, pass selling_uom to the from_uom and pass standard_uom to the to_uom.
- Set promo_retail = promo_selling_retail / convert factor.

Update_item_loc():



This function will do an array update against the item_loc table using the item_loc array passed where the records rowid in the item_loc table equal the rowids in the item_loc array. If the Batch with Online Users indicator is 'Y', it will also delete the records that were processed successfully from the prmpcupd_item_loc_temp table.

Final():

This function will call retek_close to perform the restart/recovery closing logic, as well as the last commit of the database changes.

Check_lock():

This function will acquire lock on item_loc records that are being processed.

VIII. Scheduling Considerations

Processing Cycle: Phase 3

Pro-Processing: Prmext.pc

Post-Processing:

Threading Scheme: store

IX. Locking Strategy

Before processing a group of records, it will be locked first with the no wait clause. If this group of records includes a row that has already been locked by another application, the whole group will be skipped, a flag will be set, information about these records will be written to the batch_lock_log table, and a non-fatal error will be written in the log file. The batch will then continue processing the next group of records.

X. Restart/Recovery

The logic unit of work is promotion/location.

XI. Performance Considerations

XII. Security Considerations

XIII. Design Assumptions

This program will update the item_loc table for simple promotions, no matter if the system_options.multi_promo_ind flag is on or off.

When a promotion is changed, for example an item is added to the promotion, the promotion item status will be updated or removed after the execution of prmext.pc. In order to catch the promotion changes, this program will do a full scan for all valid promotions, and update the item_loc table with new promotion prices and the changed promotion/item/location prices.

XIV. Outstanding Design Issues



| Issue Description | Priority | Issue Log Updated? |
|-------------------|----------|--------------------|
| | | |
| | | |

XV. Appendix

Technical Design – Promotion.doc (Project X).

