

Monthly Replenishment Purge [rplprg_month]

Design Overview

The replenishment extraction programs (rplext, rext) write a number of records to REPL_RESULTS, the investment buy process writes records to IB_RESULTS and the Buyer Worksheet Form populates BUYER_WKSHT_MANUAL. These tables hold information that is relevant to replenishment processes. Over time, records on these tables become unneeded and should be cleared out. The Monthly Replenishment Purge Program goes through these tables and clears out those records that are older than a predetermined number of days and drops the partitions created in the REPL_RESULTS table. The eways ewInvAdjustToRMS, ewRecieptToRMS need to be shutdown when rplprg_month.pc is run.

Scheduling Constraints

Processing Cycle: Phase 3 (monthly).

Scheduling Diagram: Can run anytime.

Pre-Processing: N/A.

Post-Processing: N/A.

Threading Scheme: N/A.

Restart/Recovery

Logical Unit of Work (recommended Commit check points)

Driving Cursor

Because this program performs only deletes, there is no need for restart/recovery or multithreading, and there is no driving cursor. However, this program still needs an entry on RESTART_CONTROL to determine the number of records to be deleted between commits.

Program Flow

Structure Chart

N/A

Shared Modules

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

N/A.

Function Level Description

All database interactions required and error handling considerations

main

Standard Retek main.

init

This function fetches various threshold dates. These dates are the current processing date (PERIOD.vdate) minus
SYSTEM_OPTIONS.repl_results_purge_days, PERIOD.vdate minus
SYSTEM_OPTIONS.store_orders_purge_days and
SYSTEM_OPTIONS.ib_results_purge_days.
This function should also fetch the commit_max_ctr from
RESTART_CONTROL to determine how many records can be deleted between
commits.

process

This program will, in four different loops, delete from REPL_RESULTS, BUYER_WKSHT_MANUAL, STORE_ORDERS and IB_RESULTS where the repl_date, create_date, need_date and create_date are less than their associated threshold date and the rownum is less than the commit counter. The BUYER_WKSHT_MANUAL table will be purged using the replenishment results purge days threshold date. If REPL_RESULTS is partitioned, this program will drop the partitions that were created and then validate all the invalid objects of the schema owner.

If this statement processes no rows, then the table has been fully purged and the function should return successfully. Otherwise, it should commit and return to the top of the loop. The loop with intermittent commits is necessary in order to ensure that rollback segments are not in danger of overflowing.

I/O Specification

All files layouts input and output
N/A.

Technical Issues

Think about using Parallel Query in the delete.
If the client doesn't want to hold on to their results, it's probably better to remove this program from the schedule and modify repl_pre to truncate REPL_RESULTS nightly (similar to the current treatment of ORD_SUGG/F).

Testing Scenarios

Insert records onto REPL_RESULTS, BUYER_WKSHT_MANUAL, STORE_ORDERS and IB_RESULTS some with a repl_date, create_date or need_date earlier than the threshold date, some later. Run replprg_month. Those records earlier than the threshold should be deleted, while those later should remain.

Process enough records (or set the commit_max_ctr low enough) so that the program must perform multiple deletes in process ().