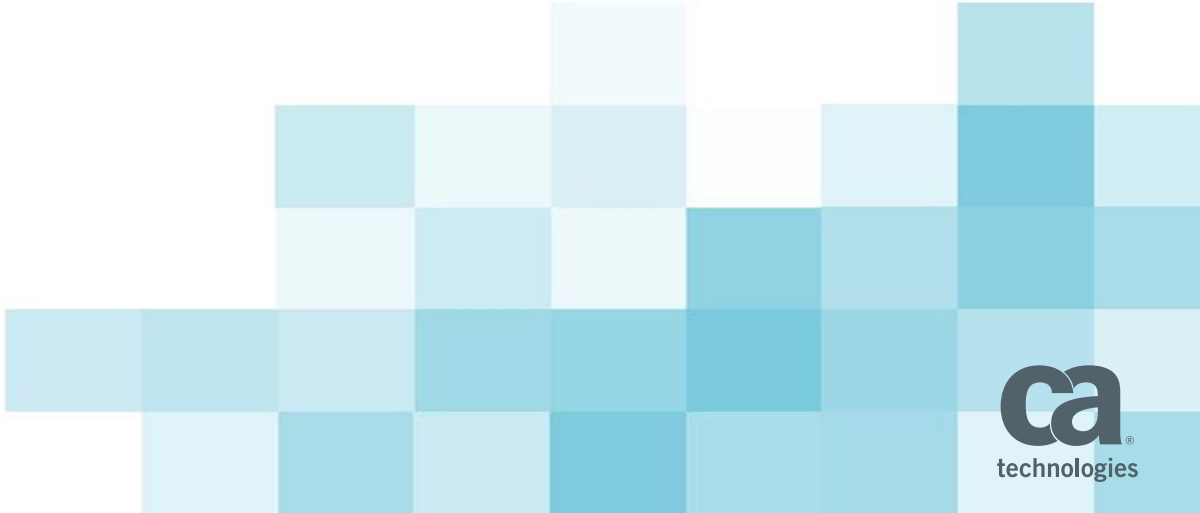
Are you getting the most value from Real Time Statistics (RTS)?

Jacek Rafalak
CA Technologies



Agenda

1. INTRODUCTION
2. WHAT DATA IS AVAILABLE?
3. HOW RTS DATA IS COLLECTED
4. USING RTS TO CONTROL MAINTENANCE ACTIVITIES
5. AREAS WHERE DB2 IS NOW USING RTS – ARE YOU USING THEM?

Introduction

- RTS introduced in DB2 V7 by PTF after the GA date
  - Data was stored in a non-Catalog database
  - RTS data ~~collection~~ externalization was optional

- Significant changes in DB2 9 for z/OS (2007)
  - RTS tables moved into the Catalog
  - Data externalization became compulsory
  - In memory Control Blocks moved above the Bar

- RTS initially intended to assist Housekeeping activities
  - Best practice since DB2 9 to base these on RTS
  - Leaving the main (RUNSTATS) Catalog Statistics for the Optimizer

- DB2 now starting to use RTS for other purposes

Where is data stored and what is available?

- ***SYSIBM.SYSTABLESPACESTATS***(SYSTSTSS) for Tablespaces
- *SYSIBM.SYSINDEXSPACESTATS*(SYSTSTSS) for Indexes
- Using system-period data versioning DB2 stores in DB2 12 RTS within

  SYSIBM. SYSTABSPACESTATS_H

  SYSIBM.SYSINDEXSPACESTATS_H
- *Updated to UTS (Single Table) Tablespaces in DB2 11*
- For the following slides items in Blue exist in both tables

RTS Tables – Common and Key columns

UPDATESTATSTIME	When RTS were last updated
DBID, PSID, ISOBID	Object Identifiers
DBNAME,NAME, Partition	Identifies object
CREATOR	Index Creator
INDEXSPACE	Index space
INSTANCE	Instance of object (1 or 2)
LASTDATACHANGE	When RTS row was last updated due to data change not Utility (DB2 11)

Tablespace – General

- NACTIVE — No. of Active pages
- NPAGES — No. of distinct pages with active data
- EXTENTS — No. of extents (last dataset if >1)
- SPACE — Space allocated in Kb
- TOTALROWS — Number of rows
- DATASIZE — No. of bytes in rows (0 for LOBs)
- UNCOMPRESSED-DATASIZE — Always set to 0 – shame!

Indexspace - General

- NLEVELS — No. of levels in the index tree
- NPAGES — No. of pages with only pseudo-deleted entries
  Note the difference in meaning from Tablespaces!
- NACTIVE — No. of Active (pre-formatted) pages
- NLEAF — No. of Leaf pages
- TOTALENTRIES — Including duplicate Index entries
- LASTUSED — Last time Index used to access data
  Includes RI related checks
  Only updated once per DPSI
  See also REORGINDEXACCESS later

Last execution times for Utilities

- LOADRLASTTIME — Last LOAD REPLACE
- REORGLASTTIME — Last REORG (or REORG INDEX)
- STATSLASTTIME — Last RUNSTATS
- COPYLASTTIME — Last COPY (Full or Incremental)
- REBUILDLASTTIME — Last REBUILD INDEX

LOAD and REORG – General counters

- Counts since the last LOAD REPLACE, REORG or object created:
  - REORGINSERTS — Inserts (including LOAD RESUME)
  - REORGUPDATES — Updates
  - REORGDELETES — Deletes
  - REORGMASSDELETE — Mass Deletes or Drop Tables from Segmented
  - UPDATESIZE — Net bytes added or removed by DML (DB2 11)

  - REORGUNCLUSTINS — Unclustered Inserts (over 16 pages distant)
  - REORGCLUSTERSENS — Reads sensitive to Clustering (DB2 10)
  - GETPAGES — The number of getpage requests for the table space since the last REORG or object creation

LOAD and REORG – Index related counters

- Counts since the last LOAD REPLACE, REORG or object created:

– REORGAPPEND-INSERT	Rows inserted at the end of the index
– REORGPSEUDO-DELETES	Entries where RID marked as deleted
– REORGNUMLEVELS	No. of Levels added since last Utility
– REORGINDEXACCESS	No. of times index was used since utility

LOAD and REORG – Disorganization Indicators

- Since the last REORG or LOAD REPLACE
- At the Tablespace level:
  - REORGNEARINDREF Overflow records near Pointer record
  - REORGFARINDREF Overflow records away from Pointer record

  *Threshold is SEGSIZE*2 pages or 16 if non-segmented*

  Effectively how likely a page is to be available after a List Prefetch
- At the Indexspace level:
  - REORGLEAFNEAR Net leaf pages close to previous pages
  - REORGLEAFFAR Net leaf pages away from previous

  Counters are incremented or reduced when leaf splits occur or leaf pages are deleted

LOAD and REORG – More Specialized Counters

- Counts since the last LOAD REPLACE, REORG or object created:
  - REORGDISORGLOB — No. of LOBs inserted but not perfectly “chunked”
  - REORGSCANACCESS — Accesses using Tablespace Scan
  - REORGHASHACCESS — Accesses using Hash key
  - HASHLASTUSED — Last time the Hash key was used

RUNSTATS counters

- Counts since the last RUNSTATS or object created:
  - STATSINSERTS — Inserts (including LOAD RESUME)
  - STATSUPDATES — Updates
  - STATSDELETES — Deletes
  - STATSMASSDELETE — Mass Deletes or Drop Tables from Segmented TS

COPY counters

- Counts since the last COPY (Full or Incremental) or object created:
  - COPYUPDATEDPAGES — Pages updated since the last Imagecopy
  - COPYCHANGES — Number of Update DML operations. Includes Rows loaded with LOAD RESUME
  - COPYUPDATELRSN — LRSN/RBA of first data change
  - COPYUPDATETIME — Timestamp of first data change

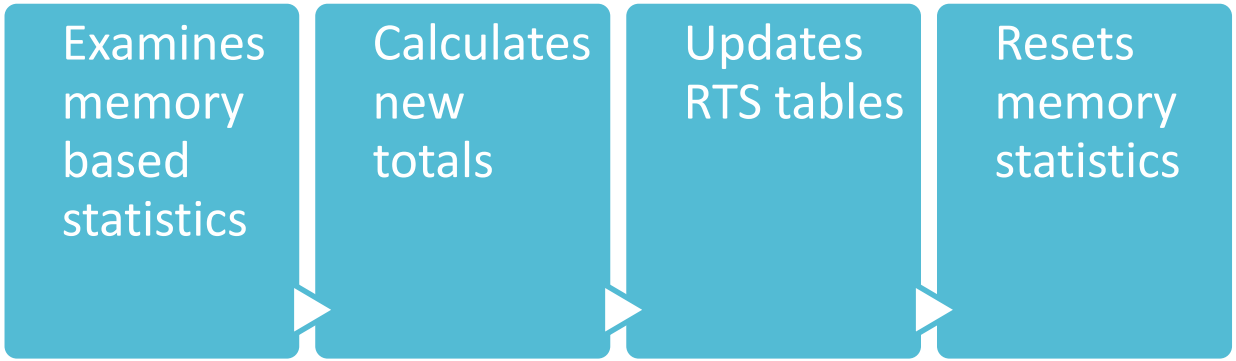
Information about the Disk Subsystem

- DRIVETYPE — HDD or SDD (Solid State)
- LPFACILITY — Disk Controller supports High Performance List Prefetch

- *Warning:* As at time of writing DRIVETYPE is only populated correctly for IBM DASD. For non-IBM devices the data may well be inconsistent.

How RTS data collection works

- DB2 collects information (statistics) in memory about all objects and indexes at the partition level
  - Stored in DBM1 address space, above the Bar since DB2 9
- RTS is simply the process of DB2 externalizing this data on a regular basis
- RTS Externalization managed by an Asynchronous Task



When does data get externalized?

- Normal driver is DSNZPARM STATSINT – defaults to 30 minutes
- Start or Stop Database
- DB2 shutdown
- -ACCESS DATABASE(xxx) MODE(STATS) – available since DB2 11
- Utilities cause some RTS data to be reset
  - REORG, RUNSTATS, LOAD, COPY, BACKUP SYSTEM
- RTS designed to avoid holding up DB2 operations
  - If something goes wrong the RTS statistics are usually lost
  - DB2 will set the RTS values to NULL if it can but not always possible
  - RTS values can be reset by executing a utility on the object

System-period data versioning for RTS (DB2 12)

SYS_START	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW BEGIN	A timestamp value that corresponds to the start time that is associated with the most recent transaction.
SYS_END	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS ROW END	A timestamp value that corresponds to when the row is deleted from the system-period temporal table.
TRANS_START	TIMESTAMP(12) NOT NULL GENERATED ALWAYS AS TRANSACTION START ID	A unique timestamp value per transaction or the null value.

RTS in Data Sharing 1/2

- Members collect & manage their own in memory Statistics
  - STATSINT is a subsystem level setting
- When these are written out the Catalog tables are updated with the values for that member
- Some utilities and operations can cross invalidate the in-memory statistics of other members in the group
  - RUNSTATS, REORG, LOAD, BACKUP SYSTEM
  - Any Operation which resets pages to empty such as Mass Delete or DROP Table
  - This forces other members to externalize their statistics
  - If update fails the operation continues but RTS are set to NULL
- Due to timing of data being externalized RTS can be negative in a Data Sharing environment

What operations update in-memory statistics?

- Update DML modifies the relevant in-memory Statistics
  - INSERT, UPDATE, MERGE, DELETE, TRUNCATE
- ROLLBACK will cause Statistics to be updated again
  - Using the reverse of the original operation
- COPY, LOAD, REORG, REBUILD INDEX and RUNSTATS all modify the relevant RTS entries
  - As you would expect – resetting the relevant counters
  - See the Managing Performance Manuals for full details
- RECOVER can also update RTS
  - TOCURRENT does not as in-memory settings still relevant
  - PIT based recoveries set Utility Counters to NULL and the Space based RTS settings to their correct values

CREATE and DROP objects

- CREATE TABLESPACE or INDEX
  - Before DB2 9 you needed to run a Utility to prime the data
  - Good news! No longer necessary
  - DB2 creates the relevant RTS rows and populates them

- DROP object
  - Generally RTS rows are deleted by DB2
  - But if RTS database is not available the object is still dropped
  - You would need to DELETE the data manually
  - If OBIDs are re-used by a new object the data is reset by DB2

Using Service aids or non-DB2 utilities

- 3rd party utilities generally update RTS values
- Service aids such as DSN1COPY or non-DB2 utilities such as DFSMS are not RTS aware
- Recommended Action:
  1. Stop the object to externalize the Statistics
  2. Invoke the utility
  3. Update RTS with correct settings and zero relevant utility counters
  4. Restart the object
- Alternative (dirty) method to set RTS settings to NULL
  - Most users of RTS will assume RTS data invalid but check!

Performance and Memory implications

- Minimal impact on Performance
  - DB2 is collecting this data internally anyway
  - The only cost is externalizing this to the Catalog
  - That only happens twice an hour by default and is cheap
- Memory requirements
  - 200 or so bytes per open partition (150 in DB2 9)
  - Data stored above the Bar since DB2 9
  - DSMAX limit is currently 200,000 but most customers much lower
  - A 100K dataset site might be looking at 20Mb

Accuracy of RTS Data

- Generally very accurate – particularly since DB2 9
  - Remember only get updated at STATSINT so an approximation
  - DB2 11 helps with –ACCESS DATABASE … MODE(STATS)
- RTS data may be missing if you have migrated from DB2 V8 and have never run a REORG on an object
- RTS data may become inaccurate
  - If objects get placed into restrictive states
  - After a Utility restart
  - After Notify failures in Data sharing
- Failure to update RTS will not cause DB2 operations to fail
  - Can result in RTS data being inaccurate or counters to be NULL

Creating a Performance database using RTS data

- "*RTS Serves as a surrogate for Performance Trending*"
  - Jim Dee – IDUG Presentation Philadelphia 2015
- Good idea to save RTS data on a regular basis
- But RTS values are reset when utilities are executed
  - Timing when to save can be hard
- Daily fine if you have a traditional Housekeeping window
  - But more and more customers don't have these
- Add a step before utility to save data about to be overwritten?
- A combination is likely to give you the most valuable data
  - Using the dates in the RTS tables to avoid saving data twice

When to organize Tablespaces

- Best practice is to use DSNACCOX (see later slide)

- Managing Performance Manual recommendations:
  - REORGUNCLUSTINS/TOTALROWS > 10% if not random access
  - (REORGNEARINDREF+REORGFARINDREF)/TOTALROWS > 5% (DS)
  - (REORGNEARINDREF+REORGFARINDREF)/TOTALROWS > 10% (non-DS)
  - REORGINSERTS/TOTALROWS > 25%
  - REORGDELETES/TOTALROWS > 25%
  - REORGDISORGLOB/TOTALROWS > 50%
  - EXTENTS > 254
  - SPACE > 2*(DATASIZE * 1024)
  - REORGMASSDELETE > 0
  - REORGCLUSTERSENS > 0 (Beware NULL after migration from DB2 10)
  - Tablespace in AREO* or any Index in ARDBP status

When to organize Indexes

- Best practice is to use DSNACCOX (see later slide)
- Managing Performance Manual recommendations:
  - REORGPSEUDODELETES/TOTALENTRIES > 10% (non-DS)
  - REORGPSEUDODELETES/TOTALENTRIES > 5% (Data Sharing)
  - REORGINSERTS/TOTALENTRIES > 25%
  - REORGDELETES/TOTALENTRIES > 25%
  - REORGAPPENDINSERT/TOTALENTRIES > 20%
  - EXTENTS > 254
  - Object in AREO* or ARDBP after an ALTER Statement
- Does not seem to look at leaf pages getting disorganized?
  - RTS REORGLEAFNEAR & REORGLEAFFAR are not checked?
- Also may be worth looking at Getpage increases

DSNACCOX

- IBM provided Stored procedure
- DSNACCOX replaced the old DSNACCOR in DB2 9
  - The old one used Catalog Statistics rather than RTS
- Implements Recommended Best Practices for Housekeeping
- You can invoke DSNACCOX checking for all or some utilities
  - Covers RUNSTATS, COPY and REORG
- Automatically kept up to date with best practice
  - E.g. PI22121 in November 2014 updated DSNACCOX to handle REORGCLUSTERSENS being Null after an upgrade to DB2 10
- The next slide is a sample to show how much effort it saves!

Sample DSNACCOX Query for Tablespace REORG

(REORGLASTTIME IS NULL AND LOADRLASTTIME IS NULL) OR (NACTIVE IS NULL OR NACTIVE > 5) AND

(((REORGINSERTS·100)/TOTALROWS>*RRTInsertsPct*) AND REORGINSERTS>*RRTInsertsAbs*) OR

(((REORGDELETE·100)/TOTALROWS>*RRTDeletesPct*) AND REORGDELETE>*RRTDeletesAbs*) OR (REORGCLUSTERSENS

> 0 AND (REORGUNCLUSTINS·100)/TOTALROWS>*RRTUnclustInsPct*) OR

(REORGDISORGLOB·100)/TOTALROWS>*RRTDisorgLOBPct* OR (SPACE·1024)/DATASIZE>*RRTDataSpaceRat* OR

((REORGNEARINDREF+REORGFARINDREF)·100)/TOTALROWS>*RRTIndRefLimit* OR

REORGMASSDELETE>*RRTMassDelLimit* OR EXTENTS>ExtentLimit)) OR ((QueryType='REORG' OR QueryType='ALL') AND

ObjectType='ALL'1 AND overflow index for hash access is used2, and (overflow index TOTALENTRY x 100) / TOTALROWS >

RRTHashOvrFlwRatio)) OR ((QueryType='RESTRICT' OR QueryType='ALL' OR *QueryType*='REORG') AND (ObjectType='TS'

OR ObjectType='ALL') AND **The table space is in advisory or informational reorg pending status**)

Optimizer usage of RTS

- From DB2 10 the Optimizer uses RTS for Table/Index Stats if:
  - Table has CARDF=0
  - Sum of CARDF for qualified Partitions is 0
  - Table has no Stats and is VOLATILE or below NPGTHRSH DSNZPARM

- Why doesn't Optimizer use RTS more widely?
  - RTS only track basic Table and Index Statistics
  - Optimizer really needs more detailed column based Statistics
  - RTS values always changing so Access path liable to change too often

New in DB2 11 – Optimizer Statistics Feedback

- Optimizer provides feedback on missing or conflicting Stats
  - Using new SYSIBM.SYSSTATFEEDBACK table for BIND, REBIND, PREPARE
  - Using DSN_STAT_FEEDBACK Plan Table for EXPLAIN

- You can use these to execute the necessary RUNSTATS
  - Up to the User or a Tool to generate the necessary syntax

- DB2 is only saying a Statistic might have been used
  - No guarantee it would be used or the Access path would change

- Some examples of REASONS
  - BASIC Basic Statistics are missing
  - CONFLICT Conflict between Table & Index statistics
  - LOWCARD Columns Cardinality Low (Likely Skew)

How Utilities use RTS

- RTS values are used to size Dynamic SORTWORK datasets
  - CHECK or REBUILD INDEX
  - REORG TABLESPACE
  - RUNSTATS using COLGROUP
  - See next slide for more details
- COPY FULL NO and COPY CHANGELIMIT in DB2 11
  - Use RTS rather than Spacemap to determine whether Copy needed
  - Copy dataset is not allocated if no pages have changed
  - SYSCOPY entry is not created
  - DB2 used to allocate an empty COPY and register it

SORTNUM avoidance (DB2 10)

- Decides who dynamically allocates SORTWORK datasets and whether SORTNUM in Syntax is used or not
  - Choice lies between the Utility and the Sort package

UTSORTAL	IGNSORTN	SORTNUM	Who allocates SORTWORK
YES	NO	YES	SORT using SORTNUM and RTS
YES	NO	NO	Utility using RTS if not in JCL Minimizes number for parallelism
YES	YES	Any	Utility using RTS if not in JCL SORTNUM is ignored
NO	Any	Any	SORT using SORTNUM & SORTDEVT or if none the SORT defaults

PCTFREE FOR UPDATE (DB2 11)

- DB2 11 introduces the PCTFREE FOR UPDATE clause
- Reserves space in Page so objects have space to expand
  - Useful for VARCHAR or Compressed objects
- Prevents fragmentation and reduces need for REORG
- FOR UPDATE -1 uses RTS to determine the setting
  - Starts with 5%
  - Adjusted using RTS at REORG time
- DSNZPARM PCTFREE_UPD sets your system default
  - 0 to 99 or AUTO
  - Default shipped with DB2 is 0

Some other areas where RTS are relevant or useful

- Autonomic cleanup of Pseudo-deleted index entries
  - Asynchronous tasks use RTS to identify indexes requiring action
- Archiving data
  - Partitions set into PRO state when you Archive them in HPSS
  - Good idea to check whether they are ever updated using RTS
  - Transparent Archiving less disruptive but still involves a UNION ALL
- ACCEL_LOAD_TABLES Stored Procedure
  - Checks whether Table has changed since it was last Loaded

CA World 2017 Call for Speakers is Open!

CA World is coming. Are you ready?

5 incredible days, over 400 empowering sessions, 70 networking opportunities, 200 customer speakers, thousands of attendees from the CA community, 1 amazing concert and you.

Share your experiences and amazing session ideas about transformative projects, compelling use cases, and best practices for thriving in the digital future

Learn more: **ca.com/caworld**

ca technologies

Jacek Rafalak
Principal Product Owner
Jacek.Rafalak@ca.com




ca
technologies