Performance Automation in DB2
LUW

Jim Wankowski
DB2 Product Manager
Quest Software
Agenda

- Understanding Monitoring Methods
- Memory Overview
  - Key memory areas
  - Performance implications
- Autonomics
- Self Tuning Memory Manager
- Space Management
  - Overview
  - Automatic space
- Maintenance
  - Maintenance policies
  - RUNSTATS
    - Automatic stats
    - Real-time statistics
  - REORGANIZATION
    - Automatic reorganization
Instance Monitoring

All aspects of the DB2 instance need to be monitored

– Take a look at the big picture
  • Think of DB2 as an ecosystem

– Do not tune for the sake of tuning!
  • Where are your bottlenecks?
Monitoring Methods

• **Snapshot Monitor**
  - Show status of database for an instant in time
    - Sort
    - Locks
    - Table activity
    - BP activity
    - UOW
    - SQL
  - Low overhead (~5%)

• **Event Monitor**
  - Historical collection of data
  - More overhead (~10-20%)
  - Main focus on application statistics
    - Tables, Deadlocks, Tablespaces, BP, connections, Statements, Transactions
Memory Management

- Catalog Cache
- Package Cache
- Sort Heap
- Buffer Pool
- Locklist

Minimizing the amount of disk access should be a key performance objective
Memory Usage

Catalog Cache

- Minimizes I/O against catalog
- Contains
  - Table descriptor information
    - Used when compiling SQL
  - Database Authorization information
    - SYSDBAUTH
    - Execute privileges for routines
  - Real-time statistics cache (9.5)

Package Cache

- Minimizes I/O against catalog
  - Loading packages
  - Having to prepare Dynamic SQL
Performance Implications

Catalog Cache

• Increased bind times
• Increased compile times
• Increased time to check DB and execution privileges
• Consider increasing size if real-time stats being used

Package Cache

• Slower response time with Dynamic SQL
What to monitor

- Catalog Cache Overflows
  - Cat_cache_overflows
- Catalog Cache lookups
  - Cat_cache_lookups
- Catalog Cache Inserts
  - Cat_cache_inserts

- Package cache overflows
  - Pkg_cache_num_overflows
- Package cache lookups
  - Pkg_cache_lookups
- Package cache inserts
  - Pkg_cache_inserts
- Package cache high water mark
  - pkg_cache_size_top
Sorting

Sort Heap

• Number of pages available for private or shared sorts
  – Used by Optimizer for determining access paths
    • Sorting
    • Hash Joins
    • Index ANDing
Optimizing Sort

• **Avoid Sort Overflow**
  – If *sortheap* too small, sort will overflow into temp database tables
  – *Sortheap* = Average total sortheap for each active database
  – *Sheapthres* = Average number of active sorts x the average size of the sort heap

• **Avoid Non-Piped Sorts**
  – If sorted information must be stored in a temporary table vs. memory (*sortheap*)
  – Determined at optimization
Optimizing Sort

- Proper indexing to minimize sorting
- Avoid ORDER by, GROUP By, DISTINCT
- Avoid sorting VARCHARs
- Only select required columns
- Rebind applications after changing sortheap
What to Monitor

- Sort Heap Overflow
  - `Sortheap_allocated`
- Active Sorts
  - `Active_sorts`
- Active Hash Joins
  - `Active_hash_joins`
Bufferpools

• Cache table and index data for future access

• IBMDEFAULTPP automatically created with database
  – Additional pools created with DDL
Effective Use of Bufferpools

Separate bufferpools for:

- Sequentially scanned tables
- Temporary table spaces
- Small frequently updated tables
- Small read-only tables
- Large tables w/random access
What to Monitor

- **Overall hit ratio**
  - Total # data/IX reads by BP
- **Data hit rate**
- **Index hit rate**
- **Asynchronous page cleaners**
  - *Num_iocleaners*

Get Snapshot for Bufferpools on DB
Locklist

Amount of storage allocated to a database for locking

Possible Performance Implications:

- Lock Escalations
  - Decrease in concurrency
  - Degradation of performance due to lock waits
- Deadlocking
- SQLCODE -912
  - Maximum # locks reached in database

How to avoid:

- Frequent COMMITs
- Consider using LOCK TABLE for application performing large number of updates
- Use CURSOR STABILITY
What to Monitor

- Lock Escalations
  - `lock_escals`
- Maximum Locks
  - `Maxlocks`
Memory Management Challenge

• Optimizing memory configurations is a complex task
  – LUW databases are highly tunable
  – Large number of configuration parameters
  – Requires high skill level
  – Can be time consuming trial and error process

• Workloads are not always consistent
  – Can change significantly throughout the month or day
  – Static configurations may not be optimal when workloads shift

• Optimal performance can only be achieved by dynamically altering memory due to workload shifts
Autonomic Computing

• Began with V8.1 of DB2 LUW
  – Dynamic memory configuration
• Designed to ease the burden of DBAs
  – provide self-monitoring, self-diagnosing, and self-managing capabilities to a DBMS
• Autonomic features include:
  – Self tuning memory
  – Automatic storage
  – Automatic statistics
  – Automatic reorg
  – Automatic compression dictionary creation
  – Automatic database backup
Self Tuning Memory Manager - STMM

• **Introduced in V9.1**
  – Automatically adapts memory to current workloads
    • Tuning frequency based on workload
  – Manages main database memory parameters
    • Bufferpools
    • Package cache \( (pckcachesz) \)
    • Locklist \( (maxlocks, locklist) \)
    • Sort Heap \( (sortheap, sheapthres, sheapthres\_shr) \)

• **Enhanced with V9.5**
  – Instance memory \( (instance\_memory) \)
    • Partitioned databases
  – Database memory \( (dbheap) \)
  – Application memory \( (appl\_memory) \)
Database Memory Consumption

• Database_memory set to AUTOMATIC (Default)
  – The database will take memory from operating system as needed

• Database_memory set to specific value
  – Provides fixed limit for maximum amount of memory a database can use
Enabling STMM

• On by default for new databases in 9.1
• SELF_TUNING_MEM = ON
• At least 2 parms must be set to automatic for self tuning to occur
  – Instance memory
    • Automatic by default
  – Application memory
    • Automatic by default
  – Database shared memory
    • Database_shared_memory
  – Package cache
    • Pckcachesz
  – Sort
    • Shepathres_shr, sortheap
  – Bufferpools
    • Set buffer pool size to AUTOMATIC
  – Locklist
    • locklist and maxlocks
Monitoring STMM

- **SNAPSHOT**
- **GET DB CFG**
- STMM Log files contain info at tuning intervals

```
2009-02-11 15:19:52.599000-360 11M1040  LEVEL: EVENT
PID  : 3976   TID  : 5120   PROC : db2syscs.exe
INSTANCE: DB2   MODE : 000   DB : TOOLSDB
APPIDL : 0-272   APPID: “LOCAL.DB2.090211131957
AUTHID : DB2ADMIN
FUNCTION: DB2 UDB, RAS/PD component, pdLogInternal, probe:120
START  : New Diagnostic Log File
DATA #1 : Build Level, 128 bytes
Instance "DB2" uses "32" bits and DB2 code release "SQL09013"
with level identifier "01040107".
Informational tokens are "DB2 v9.1.300.257", "s070719", "WR21392", Fix Pack "3".
DATA #2 : System Info, 1564 bytes
System: WIN32_NT CHI203778 Service Pack 3 5.1 x86 Family 6, model 13, stepping 8
CPU: total:1 online:1 cores per socket:1 threading degree per core:1
Physical Memory(MB): total:2047 free:1005 available:1005
Virtual Memory(MB): total:3940 free:3612
Swap Memory(MB): total:1893 free:2607
Information in this record is only valid at the time when this file was
created (see this record’s time stamp)
```
Operational Details

• **Good Candidates for STMM**
  – Box has lots of free memory
  – Multiple DBs on same machine
  – Workload changes throughout the day
  – Inexperienced DBAs
  – Multiple bufferpools

• **Poor Candidates for STMM**
  – Databases with constrained memory
    • Heavily used server
    • `Database_memory` set low
  – Volatile workloads
    • Constantly changing workloads cannot be tuned effectively
    • Out of memory conditions can still occur
  – Static workload throughout the day
Physical Design Considerations

• **1 type of Tablespace**
  – 3 Categories
    • Regular
    • Temporary
    • Large

• **Extents**
  – A unit of grouped pages
    • 2 – 256 pages
    • Similar to SEGSIZE in z/OS

• **2 Allocation Methods**
  – **SMS** – System Managed Space
    • Directory – SMS only
  – **DMS** – Database Managed Space
    • File
      – Auto-Resize
    • Device
  – **Automatic**
    • 9.5
Tablespaces: DB2 LUW

- **System Managed**
  - No finite storage specified
  - Operating System’s file manager allocates space as needed
  - Good for small tables
  - Cannot Add/delete containers after creation

- **Database Managed**
  - Space is pre-allocated
  - Better suited for large tables
  - LOBs must be DMS
  - ALTER to add containers

<table>
<thead>
<tr>
<th>Feature</th>
<th>SMS</th>
<th>DMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add containers to TS</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Separate indexes from data</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Space allocated as needed</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>High performance in heavy OLTP</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>High performance in decision support</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Ease of administration for small tables</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Flexibility of administration</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>File or device containers</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
Automated Space Management

• Auto-Resize
  – Introduced in 8.2 FP9
  – DMS file tablespace only
  – Automatically increases container size

  – **SYNTAX**
    • AUTORESIZE YES
    • INCREASESIZE
      – How much to extend container
        » Either fixed amount or %
    • MAXSIZE – Maximum allowable size for tablespace
  – Benefits of DMS performance + SMS space manageability
Automatic Storage

- **DB2 LUW 9.5**
- **Database**
  - New storage pool at DB level
  - Tablespace storage path defined when database created
- **Tablespace**
  - MANAGED BY AUTOMATIC STORAGE clause
    - Tablespace allocation managed at the DB level
      - No containers specified
      - Containers automatically allocated based on database storage paths
      - Automatically extends containers
    - DMS structure
Maintenance

• **Automated Maintenance**
  – Maintenance Policies

• **Statistics Collection**
  – Why we need statistics?
  – Automatic statistics
  – Statistics Profiles
  – Real-Time Statistics

• **Reorganization**
  – What causes fragmentation
  – Automatic reorg
Automated Maintenance

- Enabled with DB config parameter
  - auto_maint
  - Auto_tbl_maint
  - Auto_runstats
  - Auto_stmt_stats
  - Auto_stats_prof
  - Auto_prof_update
  - Auto_reorg

```
Automatic maintenance
Automatic database backup
Automatic table maintenance
Automatic runstats
Automatic statistics profiling
Automatic profile updates
Automatic reorganization

(AUTO_MAINT) = ON
(AUTO_DB_BACKUP) = OFF
(AUTO_TBL_MAINT) = ON
(AUTO_RUNSTATS) = ON
(AUTO_STATS_PROF) = OFF
(AUTO_PROF_UPD) = OFF
(AUTO_REORG) = OFF

GET DATABASE CONFIGURATION
```
Maintenance Policies

• Controls behavior of automatic statistics, and reorgs
  – XML document
    • Which objects to run automatic maintenance on
    • Controls utility settings
    • Controls maintenance windows
  – New stored procedures in 9.5 for managing maintenance polices
    • AUTOMAINT_SET_POLICY
    • AUTOMAINT_SET_POLICYFILE
    • AUTOMAINT_GET_POLICY
    • AUTOMAINT_GET_POLICYFILE
  – Sample policies
    • SQLLIB/samples/automaintcfg
Statistics

Accurate statistics are a critical factor for performance monitoring and tuning

RUNSTATS provides statistical information for:

1. Optimization of SQL
2. Monitoring status of objects
Statistics and Optimization

• Accurate statistics are the optimizers “Brain Food”
• Bad access paths because of
  – Incorrect cardinality
  – Invalid correlation statistics for JOINS
  – Inaccurate cluster ratios
  – Incorrect data volumes
Automatic Statistics Collection

• Introduced in DB2 LUW V8.2
• Background process
  – evaluates table activity at regular intervals of 2 hours
  – Statistics collected asynchronously
• Throttled to minimize performance impact
  – Low priority locks
• Statistics Profile controls what statistics are collected
• Maintenance Profiles
  – Control when automatic statistics are allowed
Statistic Profiles

- Identifies stats to be collected for a particular table
  - Table
  - Index
  - Distribution

- 2 ways to create:
  - Generated by RUNSTATS
    - SET PROFILE
    - SET PROFILE ONLY
  - Automatic Profiling
    - Statistics profile determined based on database activity
      - Only collect stats relevant to a particular workload
      - Best suited for large complex queries
      - Can incur overhead
    - AUTO_STATS_PROF
      - Activates collection of query activity
      - Collected into query feedback warehouse
    - AUTO_PROF_UPD
      - Activates automatic profiling

- Stored as CLOB
  - SELECT STATISTICS_PROFILE from SYSIBM.SYSTABLES
Query Feedback Warehouse

- Required for automatic statistics profiling
- Consists of 5 tables under SYSTOOLS schema
  1. OPT_FEEDBACK_QUERY
  2. OPT_FEEDBACK_PREDICATE
  3. OPT_FEEDBACK_PREDICATE_COLUMN
  4. OPT_FEEDBACK_RANKING
  5. OPT_FEEDBACK_RANKING_COLUMN
- Created by stored procedure
  - SYSINSTALLOBJECTS
Real-Time Statistics

- DB2 9.5
- Automatically scheduled during optimal maintenance window
  - Maintenance Policy Definition
- Provides better accuracy than Automatic statistics
  - Collected at run time
  - Optimizer will determine if current stats are accurate and update if necessary
  - Statistics may use sampling
  - Stats collected synchronously
    - limited to 5 seconds
    - Not stored in DB2 catalog
      - Statistics cache
      - Can be shared with other queries
    - Background task initiated if > 5 secs (Asynchronous)
Synchronous vs. Asynchronous

• Automatic statistics collection still occurs at 2 hour intervals
• Real-time statistics may initiate asynchronous collection
  – Synchronous time has been exceeded
  – Not enough I,U,D activity to require synchronous collection
  – Statistics were fabricated
Statistics Fabrication

- Used when optimizer determines RUNSTATS to be too expensive
- Statistics are derived vs. collected
  - Based on # pages in table
  - Page size
  - Average row length
- Uses meta-data collected by index manager
  - CARD
  - FULLKEYCARD
Monitoring Real-Time statistics

- **STATS_CACHE_SIZE**
  - Current size of statistics cache
- **STATS_FABRICATIONS**
  - Total number of statistics fabrications performed by RTS
- **SYNC_RUNSTATS**
  - Total number of synchronous RUNSTATS triggered by RTS
- **ASYNC_RUNSTATS**
  - Total number of asynchronous RUNSTATS performed by RTS
- **STATS_FABRICATE_TIME**
  - Time spent on statistics fabrication
- **SYNC_RUNSTATS_TIME**
  - Total time of synchronous RUNSTATS activities
Automatic statistics Limitations

- **Automatic or Real-Time stats are not available**
  - For tables whose stats have been manually updated
  - Statistical Views
  - Tables marked as VOLATILE
Reorg

• What causes fragmentation?
  – Insert/Update
  – How to minimize fragmentation
    • Check PCTFREE
    • Clustering indexes with PCTFREE
    • Sort data before loading
  – VARCHAR fields being updated
Monitoring for TS and Table REORGs

- **Cluster Ratio < 90%**
  - SYSCAT.INDEXES
    - CLUSTERRATIO

- **Overflow of Rows**
  - SYSSTAT.TABLES
    - OVERFLOW

- **Fetch Statistics**
  - SYSCAT.INDEXES
    - Small # of
      - AVERAGE_SEQUENCE_FETCH_PAGES
    - Growth of AVERAGE_RANDOM_FETCH_PAGES

- **Empty Pages**
  - SYSCAT.TABLES
    - FPAGES-NPAGES
Automatic Reorganization

- Requires AUTO_MAINT to be set to on
- Automates the REORGCHK process
  - Based on statistics updates
  - Determines type of reorg needed
    - Index only
    - Full table
- Table will be RO during reorg process
When Will Reorgs Occur

- **Table**
  - Overflow rows > 5% total rows
  - Table size < 68% of total space allocated
  - #pages with 0 rows > 20% of total pages

- **Index**
  - Cluster ratio < 80%
  - >50% empty space
  - # of pseudo-deleted RIDs on non-pseudo-empty pages > 20%
  - # of pseudo-empty leaf pages > 20%
Summary

• Build a strategy for how you will monitor/tune your DB2 environment
  – Try to identify most critical applications and begin focusing on those
  – Determine if STMM and/or auto maintenance are applicable
    • Turn on automatic parameters gradually and test
Jim Wankowski

Quest Software

Jim.Wankowski@quest.com

Download Presentations and whitepapers at:

WWW.QUEST.COM/DB2