DB2 FOR Z/OS:
PRACTICAL PERFORMANCE

Cristian Molaro
Independent Consultant
IBM GOLD Consultant
IBM Information Champion

DUGI, Milano – Roma 2012
Please be aware that the actual programming techniques, algorithms and all numerical parameters used in examples given in this presentation are subject to change at some future date either by a new version of DB2, a new release, a Small Programming Enhancement (SPE) or a Programming Temporary Fix (PTF).

The information contained in this presentation has not been submitted to any formal review and is distributed on an “as is” basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer’s ability to evaluate and integrate them into the customer’s operational environment. While each item may have been reviewed for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

DB2 is a trademark of International Business Machine corporation. This presentation uses many terms that are trademarks. Wherever we are aware of trademarks the name has been spelled in capitals.
Agenda

Performance and problem resolution
Overview of DB2 traces
Locking
Latching
Logging
I/O
Dynamic SQL STMTs
- Definition of Performance

Computer performance is characterized by the amount of *useful work* accomplished by a computer system compared to the time and resources used.

→ Good computer performance may involve
  - Short response time
  - High throughput
  - Low utilization of computing resources
  - High availability

→ What is your own definition of Performance?
  - IT DEPENDS
  - Greatly influenced by:
    • your business requirements
    • Your business objectives
- Problem resolution: Keep it simple!

➤ Most of the performance problems can be quickly understood
  – A well defined problem is a problem half solved
➤ Challenge 1: connect symptom with root cause
➤ Challenge 2: Keep it simple!
  – Architecture can be very complex
➤ Dynamic process
  – Continuous monitoring
  – Applications evolve
  – DB2 evolves

TIP: Build, maintain and exploit a Performance Data warehouse
- Performance Tuning strategies

→ Where to start a performance tuning campaign?
→ How much impact on performance has a system level tuning?

![Diagram showing different levels of performance impact: Application, Physical design, DB2 Subsystem, Middleware, z/OS and hardware, WLM policy.]

cristian@molaro.be ©® 2012
- DB2 Performance bottlenecks

Resource constraint situations are highly dependent on workload

100% CPU is not always a problem

**IMPORTANT:** The application performance design process should define the limiting resource assumed during the capacity sizing

Common bottlenecks

- CPU
- I/O
  - Synchronous I/O
  - Logging
- Concurrency
  - Locking
- Storage
  - Not enough Virtual storage
  - DBM1 constraint BTB
- DB2 Times analogy: trip Paris - Brussels

- CLASS 1
  - Travel total duration
  - Includes comfort stops

- CLASS 2
  - Time engine was running

- CLASS 3
  - Engine running and waiting
    - I.e. waiting in road toll

- Not accounted:
  - Engine running and waiting
  - Cannot assign waiting to a specific reason
    - I.e. unknown traffic delays
- **DB2 accounting elapsed times**

  ➔ **Class 1 elapsed time**
  - Shows the duration of the *accounting interval*
    - Depends on the application infrastructure
  - Includes time spent in DB2 and in the application
  - Also referred to as *application time*

  ➔ **Class 2 elapsed time**
  - It counts only the time spent in the DB2 address space during the accounting interval
  - It represents the sum of the times from any entry into DB2 until the corresponding exit from DB2
  - Also referred to as the *time spent in DB2*

  ➔ **Class 3 elapsed time**
  - Wait time including I/O wait time, lock and latch wait time

  ➔ **Not Accounted**: non identified time spend in DB2
- Recap of important classifications

➢ DB2 Times Classes
  - CLASS 1: response time
  - CLASS 2: time in DB2
  - CLASS 3: suspend time in DB2
  - Not accounted: (CLASS 2 – CLASS 3)
  - CLASS 7: Package level time in DB2
  - CLASS 8: Package level suspend time in DB2

➢ IFCID are grouped by CLASSES
  - I.e. ACCOUNTING CLASS 1 included IFCIDs 3 and 106
  - See hlq.SDSNIVPD(DSNWMSGS)

➢ SMF Types
  - 100: almost all of STATISTICS traces
  - 101: almost all of ACCOUNTING traces
  - 102: almost all of PERFORMANCE traces
- What classes to start?

➡️ Continuous performance monitoring
  - ACCOUNTING CLASS 1, 2 and 3 destination SMF
  - STATISTICS CLASS 1, 3, 4, 5 and 6 destination SMF

➡️ Package information
  - Start Accounting Class 7 and 8
  - Detailed Accounting information for packages: Class 10
    • See PK28561: ACCOUNTING PACKAGE DETAIL
    • Warning overhead: use when needed

**TIP:** Use zParms to automatically start DB2 traces

➡️ Detailed performance monitoring
  - Keep cost of detailed monitoring low!

➡️ See Information Center for details of IFCIDs and Classes
- Trace overhead

➤ Typical trace costs
  - Accounting Class 1,3 + Statistics 1,2,3,4 ~ 2 to 5%
  - Audit ~ < 5%
  - Performance range between 20 to 100%
  - Global ~ up to 100% or more…

➤ Performance and Global traces can be quite resource intensive

➤ Enable only the minimal trace and audit classes that you need.
  - You can enable more detailed traces only when you encounter specific performance problems

➤ Beware of possible overhead introduced by Performance monitors
- Is there a problem?

Not in DB2 time = Class 1 – Class 2

Investigate application logic

Investigate infrastructure
  – Network issues?

Investigate in DB2 time distribution
  – Class 2
  – Class 3
  – Not accounted
- Guidelines: where to start?

➔ If most time spent on non DB2 activities
  - Look for reason for bad performance outside DB2
  - Access to other databases
  - File processing
  - Etc.

➔ Otherwise look at:
  - What is the major contributor to class 2 elapsed time?
  - High 'OTHER' → System related
  - CPU wait too high
  - Excessive z/OS paging
  - VSAM problems?
  - High average Class 3 synchronous I/O suspension time
  - High Class 3 lock/latch suspension times → Application-related
  - High Class 2 CPU time → Application-related
- In DB2 Time = Accounting CLASS 2 & 3 / 7 & 8

TIP: Start here → understand where the time in DB2 is spend

→ Basic requirement!
→ Where the time in DB2 goes?
  – CPU
  – Wait
  – Unknown
→ DRDA: use package level accounting
- DB2 Class 2 Not Accounted Time

- It represents time that DB2 is unable to account for

- Also known as Not Accounted Time or Not Accounted in DB2

- It is not the same as Suspension Time!

- What is an acceptable value?
  - Non CPU constraint environments: ~ 10%
  - CPU constraints environments? A lot higher

**IMPORTANT:** DB2 Class 2 Not Accounted Time = DB2 Class 2 Elapsed time - DB2 Class 2 CPU time - DB2 Class 3 suspension time
DB2 Class 2 Not Accounted Time

- Typical value < 10% of DB2 CLASS 2

- Most common reason: waiting for CPU
  - High overall CPU utilization
  - DB2 running with Low-priority
  - Review WLM policy, reduce CPU or increase CPU capacity

- Too much detailed online tracing in performance monitor
- High MVS paging
- HSM Recall
- Monitoring locking activity

→ Statistics report
  - System-wise view
  - Scope is stats interval

→ Accounting report
  - MAX PG/ROW LOCKS HELD → good indicator of commit freq.
  - Implement COMMIT freq to keep it under ~100

### Locking Activity

<table>
<thead>
<tr>
<th>Locking Activity</th>
<th>Quantity</th>
<th>Data Sharing Locks</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEADLOCKS</td>
<td>0</td>
<td>GLB CONT. RATE (%)</td>
</tr>
<tr>
<td>TIMEOUTS</td>
<td>0</td>
<td>P/L-LOCKS XES (%)</td>
</tr>
<tr>
<td>SUSPENSIONS-LOCK</td>
<td>0</td>
<td>LOCK REQ. (P-LOCK)</td>
</tr>
<tr>
<td>SUSPENSIONS-OTHR</td>
<td>0</td>
<td>UNLOCK REQ. (P-LCK)</td>
</tr>
<tr>
<td>LOCK REQUESTS</td>
<td>62</td>
<td>CHANGE REQ. (P-LCK)</td>
</tr>
<tr>
<td>UNLOCK REQUEST</td>
<td>68</td>
<td>SYNC.XES - LOCK</td>
</tr>
<tr>
<td>LOCK ESCALAT(SH)</td>
<td>0</td>
<td>SYNC.XES - CHANGE</td>
</tr>
<tr>
<td>LOCK ESCALAT(EX)</td>
<td>0</td>
<td>SYNC.XES - UNLOCK</td>
</tr>
<tr>
<td>DRAIN REQUESTS</td>
<td>0</td>
<td>ASYN.XES-RESOURCES</td>
</tr>
<tr>
<td>CLAIM REQUESTS</td>
<td>0</td>
<td>TOTAL SUSPENDS</td>
</tr>
</tbody>
</table>

### Locking

<table>
<thead>
<tr>
<th>Locking</th>
<th>Average</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMEOUTS</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>DEADLOCKS</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>ESCAL.(SHARED)</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>ESCAL.(EXCLUS)</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>MAX PG/ROW LOCKS HELD</td>
<td>0.11</td>
<td>2</td>
</tr>
<tr>
<td>LOCK REQUEST</td>
<td>2896.06</td>
<td>52129</td>
</tr>
<tr>
<td>UNLOCK REQUEST</td>
<td>407.67</td>
<td>7338</td>
</tr>
<tr>
<td>QUERY REQUEST</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>CHANGE REQUEST</td>
<td>326.17</td>
<td>5871</td>
</tr>
<tr>
<td>OTHER REQUEST</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>TOTAL SUSPENSIONS</td>
<td>0.11</td>
<td>2</td>
</tr>
<tr>
<td>LOCK SUSPENSIONS</td>
<td>0.00</td>
<td>0</td>
</tr>
<tr>
<td>IRLM LATCH SUSPENS.</td>
<td>0.11</td>
<td>2</td>
</tr>
<tr>
<td>OTHER SUSPENS.</td>
<td>0.00</td>
<td>0</td>
</tr>
</tbody>
</table>
- Lock Tuning

**TIP:** Adopt LOCKSIZE PAGE as default. Then, in case of high DEADLOCK or TIMEOUT, consider LOCKSIZE ROW

- Performance impact?
  - Not much difference if one row lock or one page lock
  - Careful with IRLM requests!
    - Random access makes no difference
    - TableSpace scan can make a huge difference in CPU cost
- Data sharing $\Rightarrow$ additional data page P-locks acquired if LOCKSIZE ROW
- If number of locks suspensions is high, investigate:
  - Application problems
  - Incompatible workloads, i.e. OLTP and Batch

**TIP:** See redbook “DB2 9 for z/OS: Resource Serialization and Concurrency Control” SG24-4725-01
- **DB2 latches**

- **Accounting report**

<table>
<thead>
<tr>
<th>CLASS 3 SUSPENSIONS</th>
<th>AVERAGE TIME</th>
<th>AV. EVENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCK/LATCH(DB2+IRLM)</td>
<td>0.000074</td>
<td>0.89</td>
</tr>
<tr>
<td>SYNCHRON. I/O</td>
<td>0.168032</td>
<td>218.56</td>
</tr>
<tr>
<td>DATABASE I/O</td>
<td>0.020487</td>
<td>47.39</td>
</tr>
<tr>
<td>LOG WRITE I/O</td>
<td>0.147545</td>
<td>171.22</td>
</tr>
<tr>
<td>OTHER READ I/O</td>
<td>0.002769</td>
<td>4.00</td>
</tr>
<tr>
<td>OTHER WRITE I/O</td>
<td>0.000239</td>
<td>0.11</td>
</tr>
<tr>
<td>SER.TASK SWITCH</td>
<td>0.235981</td>
<td>126.50</td>
</tr>
<tr>
<td>UPDATE COMMIT</td>
<td>0.146668</td>
<td>120.06</td>
</tr>
<tr>
<td>OPEN/CLOSE</td>
<td>0.033479</td>
<td>0.61</td>
</tr>
<tr>
<td>SYSLGRNG REC</td>
<td>0.000836</td>
<td>1.17</td>
</tr>
</tbody>
</table>

DB2 10 breaks this counter down

<table>
<thead>
<tr>
<th>CLASS 3 SUSPENSIONS</th>
<th>ELAPSED TIME</th>
<th>EVENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCK/LATCH(DB2+IRLM)</td>
<td>0.000078</td>
<td>48</td>
</tr>
<tr>
<td>IRLM LOCK+LATCH</td>
<td>0.000000</td>
<td>0</td>
</tr>
<tr>
<td>DB2 LATCH</td>
<td>0.000078</td>
<td>48</td>
</tr>
<tr>
<td>SYNCHRON. I/O</td>
<td>0.001102</td>
<td>3</td>
</tr>
</tbody>
</table>

- Locking is done for DB2 by IRLM to protect sections of a database
- Latching is used for short-term serialization of DB2 resources
  - Storage or control blocks
  - Pages in the virtual buffer pool and the log write output buffer
- Very inexpensive to acquire
- Very basic in functionality (no deadlatch detection)
### DB2 Latch contention

#### Statistics report

<table>
<thead>
<tr>
<th>LATCH CNT</th>
<th>/SECOND</th>
<th>/SECOND</th>
<th>/SECOND</th>
<th>/SECOND</th>
</tr>
</thead>
<tbody>
<tr>
<td>LC01-LC04</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>LC05-LC08</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>LC09-LC12</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>LC13-LC16</td>
<td>0.00</td>
<td>0.06</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>LC17-LC20</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
</tr>
<tr>
<td>LC21-LC24</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.29</td>
</tr>
<tr>
<td>LC25-LC28</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
</tr>
<tr>
<td>LC29-LC32</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>LC254</td>
<td>0.01</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Most common:
- Log contention (LC19)
- EDM pool latch contentsions caused by a small EDM pool
- LC24 = Prefetch latch or EDM LRU chain latch
- More details on latches

- Latch Classes documented in `hlq.SDSNMACS(DSNDQVLS)`

```plaintext
QVLSLC18 DS F    /* DDF resynch list */
QVLSLC19 DS F    /* Log write */
QVLSLC20 DS F    /* System checkpoint */
QVLSLC21 DS F    /* Accounting rollup */
QVLSLC22 DS F    /* Internal checkpoint */
QVLSLC23 DS F    /* Buffer manager */
QVLSLC24 DS F    /* EDM pool or prefetch */
QVLSLC25 DS F    /* Storage manager */
QVLSLC254 DS F   /* Index lotch */
QVLSSEND DS 0C
```

- PK77514: REMOVE SERVICEABILITY LABELS FOR LATCHES

- There are many more types of latches than the number of latch classes (33)
  - Each latch class is counting for multiple types of latches
  - Run a trace for more details
- DB2 latch suspension detailed information

➔ Example:
  - The DB2 statistics latch contention section indicates a high latch contention
  - It is not immediately clear which of the underlying latches is responsible
  - Can start a detailed trace to identify the actual latch type that is causing the high latch contention
  - 51-52 Shared latch resume and shared latch wait
  - 56-57 Exclusive latch wait and exclusive latch resume

EXAMPLE:

- STA TRA(P) CLASS(30) IFCID(56,57) DEST(GTF)

➔ Very CPU intensive! < 10 seconds would be OK
➔ Note that there is no equivalent trace record for IFCID 21 (lock detail information) when dealing with DB2 latches
- DB2 Latch contention

➔ Recommendations
  - Investigate if > 10,000 latch contentions per second
  - Ignore if < 1,000 latch contentions per second

**TIP:** If high latch contention disabling Accounting Class 3 / 8 trace helps to significantly reduce CPU time as well as elapsed time

➔ Class 3 and 8 provide detail distribution of suspension times and events
➔ Overhead of Class 3 / 8 is typically about 3%, but it could increase dramatically in case of high DB2 latch suspension
➔ The overhead of the Accounting Class 3 / 8 is related to the number of events

**IMPORTANT:** Class 3 / 8 are very important for performance analysis!
- Logging

➤ Statistics report

<table>
<thead>
<tr>
<th>LOG ACTIVITY</th>
<th>QUANTITY</th>
<th>/SECOND</th>
<th>/THREAD</th>
<th>/COMMIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UNAVAILABLE OUTPUT LOG BUFF</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>OUTPUT LOG BUFFER PAGED IN</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>LOG RECORDS CREATED</td>
<td>168.2K</td>
<td>858.02</td>
<td>8854.53</td>
<td>69.69</td>
</tr>
<tr>
<td>LOG CI CREATED</td>
<td>3908.00</td>
<td>19.93</td>
<td>205.68</td>
<td>1.62</td>
</tr>
<tr>
<td>LOG WRITE I/O REQ (LOG1&amp;2)</td>
<td>14753.00</td>
<td>75.24</td>
<td>776.47</td>
<td>6.11</td>
</tr>
<tr>
<td>LOG CI WRITTEN (LOG1&amp;2)</td>
<td>18347.00</td>
<td>93.57</td>
<td>965.63</td>
<td>7.60</td>
</tr>
</tbody>
</table>

➤ UNAVAILABLE OUTPUT LOG BUFF
- Number of waits caused by an unavailable log output log buffer
- If >0 increase OUTBUFSIZE

➤ OUTPUT LOG BUFFER PAGED IN
- Number of times a log output buffer had to be paged in
- If >0 decrease OUTBUFSIZE

TIP: Logging rate [bytes/sec] = ((LOG CI WRITTEN * 4096)/(STATIME * 60))
- DB2 logging striping

<table>
<thead>
<tr>
<th>Stripes</th>
<th>Nbr of Updates</th>
<th>Log Volume (MB)</th>
<th>Elapsed (sec)</th>
<th>MB/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2000</td>
<td>128</td>
<td>7.380</td>
<td>17.36</td>
</tr>
<tr>
<td>1</td>
<td>5000</td>
<td>320</td>
<td>18.657</td>
<td>17.17</td>
</tr>
<tr>
<td>1</td>
<td>20000</td>
<td>1281</td>
<td>72.141</td>
<td>17.76</td>
</tr>
<tr>
<td>1</td>
<td>50000</td>
<td>3203</td>
<td>182.554</td>
<td>17.55</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>17.46</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2000</td>
<td>128</td>
<td>4.485</td>
<td>28.57</td>
</tr>
<tr>
<td>2</td>
<td>5000</td>
<td>320</td>
<td>12.096</td>
<td>26.48</td>
</tr>
<tr>
<td>2</td>
<td>20000</td>
<td>1281</td>
<td>48.200</td>
<td>26.58</td>
</tr>
<tr>
<td>2</td>
<td>40000</td>
<td>2563</td>
<td>87.928</td>
<td>29.14</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>27.69</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2000</td>
<td>128</td>
<td>3.897</td>
<td>32.88</td>
</tr>
<tr>
<td>3</td>
<td>5000</td>
<td>320</td>
<td>9.739</td>
<td>32.89</td>
</tr>
<tr>
<td>3</td>
<td>20000</td>
<td>1281</td>
<td>40.248</td>
<td>31.83</td>
</tr>
<tr>
<td>3</td>
<td>50000</td>
<td>3203</td>
<td>103.193</td>
<td>31.04</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>32.16</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2000</td>
<td>128</td>
<td>3.162</td>
<td>40.52</td>
</tr>
<tr>
<td>4</td>
<td>5000</td>
<td>320</td>
<td>7.441</td>
<td>43.05</td>
</tr>
<tr>
<td>4</td>
<td>20000</td>
<td>1281</td>
<td>30.147</td>
<td>42.50</td>
</tr>
<tr>
<td>4</td>
<td>40000</td>
<td>2563</td>
<td>67.007</td>
<td>38.24</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>41.08</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2000</td>
<td>128</td>
<td>3.888</td>
<td>32.95</td>
</tr>
<tr>
<td>5</td>
<td>5000</td>
<td>320</td>
<td>8.684</td>
<td>36.89</td>
</tr>
<tr>
<td>5</td>
<td>15000</td>
<td>961</td>
<td>28.486</td>
<td>33.74</td>
</tr>
<tr>
<td>5</td>
<td>50000</td>
<td>3203</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>34.53</td>
<td></td>
</tr>
</tbody>
</table>

→ Consider logging striping for high logging demanding applications
→ New I/O technology improved performance up to > 100 MB/sec
  - Recent hardware can relax the need for stripping
- DB2 COMMIT & synchronous writes

1. Write log page to Log 1
2. Write log page to Log 2
- Logging suspension time

![Graph showing logging suspension time comparison between DB2 9 and DB2 10.](image)

- **DB2 9**
- **DB2 10**

- **LOG Pages per COMMIT**

- **cristian@molaro.be © 2012**
- **I/O suspensions**

- Synchronous I/O suspensions: the most common reason for excessive wait times and long response time

- Accounting report

<table>
<thead>
<tr>
<th>CLASS 3 SUSPENSIONS</th>
<th>AVERAGE TIME</th>
<th>AV.EVENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCK/LATCH(DB2+IRLM)</td>
<td>0.000074</td>
<td>0.89</td>
</tr>
<tr>
<td>SYNCHRON. I/O</td>
<td>0.168032</td>
<td>218.56</td>
</tr>
<tr>
<td>DATABASE I/O</td>
<td>0.020487</td>
<td>47.39</td>
</tr>
<tr>
<td>LOG WRITE I/O</td>
<td>0.147545</td>
<td>171.22</td>
</tr>
<tr>
<td>OTHER READ I/O</td>
<td>0.002769</td>
<td>4.00</td>
</tr>
<tr>
<td>OTHER WRTE I/O</td>
<td>0.000239</td>
<td>0.11</td>
</tr>
</tbody>
</table>

- A single counter for Synchronous reads and writes, see BP info

<table>
<thead>
<tr>
<th>BP15</th>
<th>BPOOL ACTIVITY</th>
<th>AVERAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPOOL HIT RATIO (%)</td>
<td>97.45</td>
<td></td>
</tr>
<tr>
<td>GETPAGES</td>
<td>52922.00</td>
<td></td>
</tr>
<tr>
<td>BUFFER UPDATES</td>
<td>17351.50</td>
<td></td>
</tr>
<tr>
<td>SYNCHRONOUS WRITE</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>SYNCHRONOUS READ</td>
<td>269.00</td>
<td></td>
</tr>
<tr>
<td>SEQ. PREFETCH REQS</td>
<td>22.50</td>
<td></td>
</tr>
<tr>
<td>LIST PREFETCH REQS</td>
<td>501.00</td>
<td></td>
</tr>
<tr>
<td>DYN. PREFETCH REQS</td>
<td>1448.00</td>
<td></td>
</tr>
<tr>
<td>PAGES READ ASYNCHR.</td>
<td>1080.00</td>
<td></td>
</tr>
</tbody>
</table>
- Synchronous reads vs. synchronous writes I/O

➤ Synchronous buffer pool reads:
- One or few consecutive pages are retrieved
- Requested pages are not consecutive
- Prefetch disabled via VPSEQT=0
- Sequential Prefetch threshold (90% of VPSIZE) reached
  prefetch pages stolen before the application gets to them

➤ Synchronous buffer pool writes:
- Immediate Write threshold (97.5% of VPSIZE) reached more
  than two checkpoints have passed without the page being
  written
- Write engine not available

➤ In data sharing case, an index split forces log write for non-leaf
  pages
- **Tuning guidelines**

**IMPORTANT:** First step is to investigate origin of high Synch I/O suspension:
- High number of short suspensions?
- Low number of long suspensions?

- High number of short suspensions
  - Excessive GetPages → Use Explain to tune access path
  - Buffer pools contention → Look at BP statistics
  - Disorganized data/indexes → Check statistics and reorg / rebuild

- Low number of long suspensions, i.e. large time per suspension
  - Investigate I/O response time
  - DASD contention → PAV
  - Disk replication, PPRC or SDRF
  - Control Unit cache misses
  - CPU contention
- Dynamic SQL_STMT monitoring

**STATISTICS REPORT – LONG**

<table>
<thead>
<tr>
<th>DYNAMIC SQL_STMT</th>
<th>QUANTITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREPARE REQUESTS</td>
<td>201.00</td>
</tr>
<tr>
<td>FULL PREPARES</td>
<td>24.00</td>
</tr>
<tr>
<td>SHORT PREPARES</td>
<td>178.00</td>
</tr>
<tr>
<td>GLOBAL CACHE HIT RATIO (%)</td>
<td>88.12</td>
</tr>
<tr>
<td>IMPLICIT PREPARES</td>
<td>0.00</td>
</tr>
<tr>
<td>PREPARES AVOIDED</td>
<td>0.00</td>
</tr>
<tr>
<td>CACHE LIMIT EXCEEDED</td>
<td>0.00</td>
</tr>
<tr>
<td>PREP STMT PURGED</td>
<td>0.00</td>
</tr>
<tr>
<td>LOCAL CACHE HIT RATIO (%)</td>
<td>N/C</td>
</tr>
</tbody>
</table>

**TIP:** GDSC hit ratio should be > 90-95%

**TIP:** LDSC hit ratio should be >70%

**EDMSTMTC** determines the size (in KB) of the statement cache that is to be used by the EDM. Can be **decreased** online.
- Tracing Dynamic STMT Cache

- IFCID 316: reports on the contents of the prepared SQL in the Global Dynamic STMT cache
- IFCID 317: provides the SQL statement text from the DSC

**TIP:** use `EXPLAIN STMTCACHE ALL` to explain ALL the STMTs in the DSC

- `DSN_STATEMENT_CACHE_TABLE`
  - A row for each SQL statements in the global DSC
  - `STMT_TEXT` is a CLOB(2M) → table can grow quickly!
  - `hlq.SDSNSAMP(DSNTESC)`
  - Use SQL for performance investigation:

**EXAMPLE:**
```
SELECT STMT_ID, STAT_CPU, STAT_EXEC, STAT_GPAG, STAT_TS
FROM DSN_STATEMENT_CACHE_TABLE
ORDER BY EXPLAIN_TS DESC, STAT_CPU DESC
FETCH FIRST 5 ROWS ONLY;
```
- Tracing Dynamic STMT Cache

<table>
<thead>
<tr>
<th>STMT_ID</th>
<th>STAT_CPU</th>
<th>STAT_EXEC</th>
<th>STAT_GPAG</th>
<th>STAT_TS</th>
</tr>
</thead>
<tbody>
<tr>
<td>56</td>
<td>+0.8689610673493030E+02</td>
<td>1843</td>
<td>16587</td>
<td>2011-04-27-08.36.38.697014</td>
</tr>
<tr>
<td>50</td>
<td>+0.5469929887360217E+02</td>
<td>1157</td>
<td>10404</td>
<td>2011-04-27-08.36.38.697014</td>
</tr>
<tr>
<td>54</td>
<td>+0.1787785722321155E+02</td>
<td>1848</td>
<td>14942</td>
<td>2011-04-27-08.36.38.697014</td>
</tr>
<tr>
<td>55</td>
<td>+0.1724589921539905E+02</td>
<td>9343</td>
<td>27555</td>
<td>2011-04-27-08.36.38.697014</td>
</tr>
<tr>
<td>51</td>
<td>+0.1501635670755424E+02</td>
<td>3682</td>
<td>11046</td>
<td>2011-04-27-08.36.38.697014</td>
</tr>
</tbody>
</table>

**TIP:** use `EXPLAIN STMTCACHE STMTID #` to explain a single STMT in the DSC

**EXAMPLE:** `EXPLAIN STMTCACHE STMTID 56;`

- **PLAN_TABLE:** access path as in the Global DSC; `QUERYNO`
- **DSN_STATEMNT_TABLE:** `QUERYNO`
- **DSN_FUNCTION_TABLE:** `QUERYNO`

Extended explain tables are not populated (there is no EXPLAIN)
- Impacts of dynamic SQL monitors

- A dynamic SQL monitor adds a consistent cost of about 14% in SU
- IFCID 318 turned on adds ~3% in SU costs
- DB2 10 Literal replacement

» Dynamic SQL with literals can be re-used in the DSC
» Literals replaced with &

**EXAMPLE:**

```sql
WHERE ACCOUNT_NUMBER = 123
```

Becomes:

```sql
WHERE ACCOUNT_NUMBER = &
```

» How to enable:

- **CONCENTRATE STATEMENTS WITH LITERALS** in the PREPARE ATTRIBUTES clause
- Or set **LITERALREPLACEMENT** in the ODBC initialization file
- Or set the keyword `enableLiteralReplacement='YES'` in the JCC Driver
- **DB2 10 Literal replacement**

<table>
<thead>
<tr>
<th>DYNAMIC SQL STMT</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>REOPTIMIZATION</td>
<td>0</td>
</tr>
<tr>
<td>NOT FOUND IN CACHE</td>
<td>4</td>
</tr>
<tr>
<td>FOUND IN CACHE</td>
<td>1</td>
</tr>
<tr>
<td>IMPLICIT PREPARES</td>
<td>0</td>
</tr>
<tr>
<td>PREPARES AVOIDED</td>
<td>0</td>
</tr>
<tr>
<td>CACHE_LIMIT_EXCEEDED</td>
<td>0</td>
</tr>
<tr>
<td>PREP_STMT_PURGED</td>
<td>0</td>
</tr>
<tr>
<td>CSWL - STMTS PARSED</td>
<td>5</td>
</tr>
<tr>
<td>CSWL - LITS REPLACED</td>
<td>3</td>
</tr>
<tr>
<td>CSWL - MATCHES FOUND</td>
<td>1</td>
</tr>
<tr>
<td>CSWL - DUPLS CRETED</td>
<td>0</td>
</tr>
</tbody>
</table>

**Performance**

- Using parameter marker still provides best performance
- Biggest performance gain for repeated SQL with different literals
- Literals could provide better access path in case of data skew
- Do not forget the DB2 Address Spaces

- A complete benchmark must consider CPU in DB2 AS

- Follow these steps
  - (Optional) Switch SMF using the command /I SMF
  - Issue a MODIFY TRACE command to produce a new statistics record before starting your testing
  - As the single user of the DB2 subsystem, perform the tests
  - Issue again a MODIFY TRACE command
  - (Optional) Switch SMF using the command /I SMF
- DB2 System address spaces CPU

- Documented on STATISTICS reports

<table>
<thead>
<tr>
<th>CPU TIMES</th>
<th>TCB TIME</th>
<th>PREEMPT SRB</th>
<th>NONPREEMPT SRB</th>
<th>TOTAL TIME</th>
<th>PREEMPT IIP SRB</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM SERVICES ADDRESS SPACE</td>
<td>0.004221</td>
<td>0.000066</td>
<td>0.001967</td>
<td>0.006253</td>
<td>N/A</td>
</tr>
<tr>
<td>DATABASE SERVICES ADDRESS SPACE</td>
<td>0.000367</td>
<td>0.001281</td>
<td>0.000136</td>
<td>0.001783</td>
<td>0.000000</td>
</tr>
<tr>
<td>IRLM</td>
<td>0.000002</td>
<td>0.000000</td>
<td>0.062718</td>
<td>0.062720</td>
<td>N/A</td>
</tr>
<tr>
<td>DDF ADDRESS SPACE</td>
<td>0.005500</td>
<td>0.000000</td>
<td>0.000121</td>
<td>0.005622</td>
<td>0.000000</td>
</tr>
<tr>
<td>TOTAL</td>
<td>0.010089</td>
<td>0.001346</td>
<td>0.064943</td>
<td>0.076378</td>
<td>0.000000</td>
</tr>
</tbody>
</table>

- TCB: task control block
- SRB: service request block
- For distributed applications DDF SRB is typically high
  - Includes Accounting TCB time

**IMPORTANT:** All TCB times should be low compared to MSTR and DBM1 SRB times. IRLM SRB time should be low compared to MSTR and DBM1 SRB times
Agenda

- Performance and problem resolution
- Overview of DB2 traces
- Locking
- Latching
- Logging
- I/O
- Dynamic SQL STMTs
THANKS!

CRISTIAN MOLARO

cristian@molaro.be
@cristianmolaro