Using Solid-State Devices to Solve a DB2 INSERT Problem

Terry L. Berman
DST Systems, Inc.

March 12, 2012
Session 10842
Agenda

Case 1 – Batch INSERT SLA
  • Problem, prior remedies

Solid State Devices (SSD)
  • Technology “sweet spot”
  • Deployment
  • Case 1 results

Case 2 – CICS mass INSERTs
  • The business case

Measurement
  • Analyze performance issue for bottlenecks
  • Designate the SSD candidates
  • Measure, report results

Futures …
www.dstsystems.com

Retirement Solutions  Health Solutions

Mainframe
z196
z10’s

DB2 for z/OS  9 ➔ 10

Database Support Infrastructure

Automations
Repositories
Information delivery

IBM 8x00
DASD

SHARE in Atlanta 2012
Case 1 – Single Batch INSERT Bottleneck

Insert a large number of rows in the shortest possible time

- Retirement Solutions
- Batch calculate Dividends (month-end)
- … to be inserted into tables TRN, ATRN
- Rest of Batch has to wait
- SLA’s tied to various Batch deliverables

common business scenario

- TRN 25 min
- ATRN 80 min

Largest customer 1.2M INSERTs

250 / sec
Parallelism – (Divide and Conquer)

- Multiple jobs
  - Subdivide data
  - 8 simultaneous jobs, 8 initiators
  - Less complex vs. multi-tasking in same AS (LE)

*OK for now, but can scale to larger customers in future?*

- DB2 for z/OS lacks a parallel INSERT capability
  - As DB2 for LUW provides
INSERT processing (costs)

**INDEX**

- Maintain the B-TREE
- Enforce uniqueness
- Split if necessary

**TABLESPACE**

- Put new rows (into Cluster order)

- Space search (spacemap pages)
  - Conserve space if possible
  - Extend object if not

- Log changes under UOW
- Data Sharing integrity

- Partitioned
- Segmented, UTS PBG
- UTS PBR

- Changed pages “trickle out” asynchronously
DB2 Instrumentation (IFI Traces)  

<table>
<thead>
<tr>
<th>SMF</th>
<th>ACCTG</th>
<th>STAT</th>
<th>PERFM</th>
<th>SMF 42-6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Each DB2 thread</td>
<td>Subsystem metrics, per minute</td>
<td>Specialized traces</td>
<td>Dataset I/O</td>
</tr>
</tbody>
</table>

... exploited systematically
ACCTG breaks down elapsed times

- **CLASS 1**
  - Clock

- **CLASS 2**
  - In DB2

- **CLASS 3**
  - Suspend (30 causes)

  - CPU: 25%
    - waiting
  - Sync_Read_Wt: 60%
  - Async_Read_Wt
    - Lock / Latch
    - Commit (log sync)
    - Log Write
    - Page Latch (BP)
    - Data Sharing
  - TRN / ATRN
  - INSERT

- other INSERTs
Sync_Read Breakdown – Bufferpool Stats

- Maintaining Clustering
  - SELECT gain vs. **INSERT** pain
  - Due diligence
  - REORG (~24x7) to recluster

- APPEND removes cleanly
  - 450 / sec

Tables

- 56%  Sync_Read %  44%

Indexes

Share same BP

Which index(es) need the I/O relief?
IFCID 199 – Dataset Stats – Sync_Read

- STAT CLASS(8) – on DB2 startup (ZPARM)
- -DIS BP LSTATS information (held in Buffer Manager)
- Each open dataset, at least 1 I/O / Second

<table>
<thead>
<tr>
<th>Timestamp (STCK)</th>
<th>Each Minute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database, Space, Piece/Part</td>
<td>Dataset (“object”)</td>
</tr>
<tr>
<td># Sync_Read</td>
<td>Health check</td>
</tr>
<tr>
<td>Sync_Read Response</td>
<td>ms granularity</td>
</tr>
</tbody>
</table>

Busy subsystem main index BP 5/4/2011

IFCID 002 Subsystem BP Stats

<table>
<thead>
<tr>
<th>IFCID 199</th>
</tr>
</thead>
<tbody>
<tr>
<td>62,739,166</td>
</tr>
<tr>
<td>62,686,478</td>
</tr>
</tbody>
</table>

99.9% attributable by object

... an I/O monitor built into the engine
Which index(es) need the I/O relief?

<table>
<thead>
<tr>
<th>IX Name</th>
<th>Key len</th>
<th>Clust</th>
<th>% Sync_Read</th>
</tr>
</thead>
<tbody>
<tr>
<td>X0</td>
<td>P 12</td>
<td>NPI</td>
<td>77%</td>
</tr>
<tr>
<td>X1</td>
<td>D 50</td>
<td>Clust Part</td>
<td>~100%</td>
</tr>
<tr>
<td>X2</td>
<td>D 38</td>
<td>NPI</td>
<td>89%</td>
</tr>
<tr>
<td>X3</td>
<td>D 28</td>
<td>NPI</td>
<td>97%</td>
</tr>
</tbody>
</table>

IFCID 199

- ~0 “Dumb-IDs” in sequence
- 90% SKIP SEQUENTIAL
- ~0 “Dumb-IDs” in sequence
- 10% Dividend adjacency
What IS the troublesome X1 insert pattern?

**PERFM TRACE** *(used by Buffer Pool tools)*

<table>
<thead>
<tr>
<th>IFCID</th>
<th>Qualify by job?</th>
<th>Qualify by object?</th>
<th>Volume/overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>6, 7</td>
<td>Yes</td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>9,10</td>
<td>NO</td>
<td>NO</td>
<td>High</td>
</tr>
<tr>
<td>198</td>
<td>Yes</td>
<td>NO</td>
<td>Very high</td>
</tr>
</tbody>
</table>

**The DB2 Log**

No application remedy

DSN1LOGP SYSPRINT post-process
GETPAGE

Latencies / Costs of data access

Bufferpool (DRAM)

~1-4 msec

DASD Cache

0.25 ms

cache MISS

HDD 15K RPM

4-8 ms

Seek Disconnect

HDD 7.2K RPM

SSD

No Seek

CPC

DASD

GETPAGE

Latencies / Costs of data access

Bufferpool (DRAM)

~1-4 msec

DASD Cache

0.25 ms

cache MISS

HDD 15K RPM

4-8 ms

Seek Disconnect

HDD 7.2K RPM

SSD

No Seek

CPC

DASD
DASD Cache Hit %

- Critical in I/O Response of Sync_Read
  
- Cache algorithm uses adjacency patterns
  - Track = 12 (4K) DB2 Pages

- Less good at busy times of high I/O activity
  - Cannot designated “favored” objects

\[
\text{I/O Response} = \frac{\text{Sync\_Read\_Wt}}{\text{Sync\_Read}}
\]

For TRN / ATRN Inserts (at a Busy time)

\[
= 3-4 \text{ msec}
\]
## Solid State Drives – Flash Memory “Cells”

<table>
<thead>
<tr>
<th></th>
<th>Single-Level Cell</th>
<th>Multi-Level Cell</th>
<th>More expensive (/ bit)</th>
<th>100,000 rewrites lifetime</th>
<th>10,000 rewrites</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SLC</strong></td>
<td>1 bit</td>
<td>2 bits</td>
<td>Cheaper</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>MLC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **More reliable** vs. HDD
  - No moving parts
  - 10 years lifetime vs. 5 years
- **Less durable** vs. HDD
  - Limited number of rewrites
  - Wear out, must be replaced
- **50% power**
- **Less noise**
- **7 – 10x the cost** of HDD
IBM DS8x00 implementation of SSD

DS8000 : Introducing Solid State Drives

SLC only
More reliable than MLC

RAID 5
RAID 6 redundancy not req’d
RAID 10 performance not req’d

Error Detection / Correction
Assures accurate data
Bad blocks removed

Wear-Leveling
Even out use
Move static data to high-wear blocks

Flash Controller

Over-provisioned
up to 75% add. cells
for redundancy

64 pages

Each write uses a new block
Copies all data
No hot spots

0.8 msec response (added “enterprise” microcode)
SSD sweet spot – tuning I/O path

Ready to Access DB2 for z/OS Data on Solid-State Drives
Jeffrey Berger, Paolo Bruni – REDP-4537-00  Dec 7, 2009

Random I/O

<table>
<thead>
<tr>
<th></th>
<th>front-end seek</th>
<th>NO SEEK</th>
<th>Bottleneck</th>
<th>SSD benefit?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequential Prefetch</td>
<td>front-end throughput</td>
<td>no advantage vs. HDD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynamic Prefetch</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>List Prefetch</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Adaptive Multistream Prefetch

Hyper-PAVs

Device Adapter

SSD

HDD

Channel

zHPF

Cache

limit
Deploying SSD in ~1600GB “ranks”

Phase 1
DIV Inserts
- ATRNX1 90%
- ATRNX3 10%
- TRNX1 few%

Phase 2
Batch Peaks
- ATRNX1 7947K
- ATRNX1 7291
- ATRN 4256
- TRNX1 2529

REORG “shadows”
- Tablespace Partition(s)
- Index Partition(s)
- NPSI

Enough for worst-case simultaneous REORGs

Sync_Read (IFCID 199) surveys
Deploying SSD – SMS configuration

Virtual 3390 VOLSERs
(just like HDD)

D SMS,STORGRP(sg),LISTVOL DCOLLECT

Automatic Class Selection rules

STORCLAS SSDsc

STORGRP

“SSDdsn*” Inflexible

FILTLIST DSNAMe matching rules

SSD
Mod 3
Mod 9
Mod 27
Deploying SSD – Object by Object

CREATE STOGROUP SSDsg
  VOLUMES('**') VCAT vcatname
  STORCLAS SSDsc

-STOP DB(db) SPACE(space)
  ALTER TABLESPACE db.ts [PART p] STOGROUP SSDsg
  ALTER INDEX ixcr.ixname [PART p] STOGROUP SSDsg
-START DB(db) SPACE(space)

SSD STOGROUP
ALTER object(s)

Specify PRIQTY, SECQTY to “fit”

REORG Shrlevel Change
  Shadow (⇒ object) allocated on SsdStorclas

DFSMSdss
  Faster, but requires STOP downtime

Move
INSERT ATRN throughput

Original

APPEND

z10 → z196
M-row INSERT

SSD

ATRNX1 I/O Response

2 – 4 ms

0.8 ms
Case 2 – Mass INSERTs in OLTP

- Representative day – 4.6M transactions, 22.8M INSERTs
- Goal – Reduce DB2 Elapsed / (CICS) Transaction
- Especially given – an upcoming conversion of tables to DB2 from an older legacy database
DB2 Elapsed_Time Analysis (ACCT Class 3)

Sync_Read_Wt

78%

More typically
65%

Async_Read_Wt

12%

Other

CPU

Lock / Latch
Data Sharing
Log Write
Syslgrng
...

small
### SMF 42-6 – DFSMS DASD Dataset I/O

**MXG**

<table>
<thead>
<tr>
<th>SMF Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMFTIME</td>
<td>(60 minute interval)</td>
<td></td>
</tr>
<tr>
<td>S42JDJNM</td>
<td>JOB</td>
<td>= 'db2xDBM1'</td>
</tr>
<tr>
<td>S42DSN</td>
<td>DSNAME</td>
<td>VSAM LLDS (TS/IX)</td>
</tr>
<tr>
<td>S42DSVOL</td>
<td>VOLSER</td>
<td></td>
</tr>
<tr>
<td>S42DSIOR</td>
<td>RESPTIME</td>
<td>RESPONSE TIME (MS)</td>
</tr>
<tr>
<td>S42DSCND</td>
<td>CACHCAND</td>
<td># CACHE CANDIDATES</td>
</tr>
<tr>
<td>S42DSWCN</td>
<td>WRITCAND</td>
<td># WRITE CANDIDATES</td>
</tr>
<tr>
<td>S42DSSHTS</td>
<td>CACHHITS</td>
<td># CACHE HITS</td>
</tr>
<tr>
<td>S42DSWHI</td>
<td>WRITHITS</td>
<td># WRITE HITS</td>
</tr>
<tr>
<td>calc</td>
<td>RDHITPCT</td>
<td>READ CACHE HIT %</td>
</tr>
<tr>
<td>S42DSION</td>
<td>IOCOUNT</td>
<td># I/Os</td>
</tr>
<tr>
<td>S42SIOD</td>
<td>AVGDISMS</td>
<td>AVG I/O DISCONNECT</td>
</tr>
<tr>
<td>S42DSRDT</td>
<td>S42DSRDT</td>
<td># READS</td>
</tr>
<tr>
<td>S42DSRDD</td>
<td>S42DSRDD</td>
<td>AVG READ DISCONNECT</td>
</tr>
</tbody>
</table>

**Isolate READ (WRITE irrelevant)**

Read + Write

Read = Total - Write

**OA25559 z/OS 1.8**
Candidates – FLASHDA Service Aid


- VOLUMES with high disconnect
  - Move to SSD with zDMF and TDMF®
  - Ineffective, since datasets move with REORGs

- DATASETS with highest read disconnect
  - By DSNAME / VOLSER – not what DB2 can use

SMF 74-5

SMF 42-6

<table>
<thead>
<tr>
<th>S42DSRDT</th>
<th>#Reads Avg. Read Disc</th>
</tr>
</thead>
<tbody>
<tr>
<td>S42DSRDD</td>
<td></td>
</tr>
</tbody>
</table>

\[ \times = \text{READ\_DISC\ aggregate} \]
# Highest READ_DISC survey – Results

<table>
<thead>
<tr>
<th>TBNAME</th>
<th>IXNAME</th>
<th>PA</th>
<th>--Read Disc--</th>
<th>Cache</th>
<th>---GB Used---</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>RT</td>
<td>Pct</td>
<td>Cum</td>
<td>%Space</td>
</tr>
<tr>
<td>TAB1</td>
<td>TAB1X2</td>
<td>-</td>
<td>3.4%</td>
<td>3.4%</td>
<td>29.8%</td>
</tr>
<tr>
<td>TAB2</td>
<td>TAB2X2</td>
<td>-</td>
<td>3.1%</td>
<td>6.5%</td>
<td>28.8%</td>
</tr>
<tr>
<td>TAB3</td>
<td>TAB3X2</td>
<td>-</td>
<td>2.8%</td>
<td>9.3%</td>
<td>27.3%</td>
</tr>
<tr>
<td>TAB3</td>
<td>TAB3X7</td>
<td>-</td>
<td>2.8%</td>
<td>12.2%</td>
<td>26.6%</td>
</tr>
<tr>
<td>TAB3</td>
<td>TAB3X3</td>
<td>-</td>
<td>2.8%</td>
<td>15.0%</td>
<td>27.2%</td>
</tr>
<tr>
<td>TAB3</td>
<td>TAB3X6</td>
<td>-</td>
<td>2.8%</td>
<td>17.8%</td>
<td>25.7%</td>
</tr>
<tr>
<td>TAB4</td>
<td>TAB4X3</td>
<td>-</td>
<td>2.7%</td>
<td>20.5%</td>
<td>25.2%</td>
</tr>
<tr>
<td>TAB4</td>
<td>TAB4X2</td>
<td>-</td>
<td>2.6%</td>
<td>23.1%</td>
<td>22.9%</td>
</tr>
<tr>
<td>TAB2</td>
<td>TAB2X1</td>
<td>-</td>
<td>2.5%</td>
<td>25.6%</td>
<td>28.4%</td>
</tr>
<tr>
<td>TAB2</td>
<td>TAB2X4</td>
<td>-</td>
<td>2.5%</td>
<td>28.1%</td>
<td>28.6%</td>
</tr>
<tr>
<td>TAB5</td>
<td><em>TP</em></td>
<td>1</td>
<td>1.5%</td>
<td>29.6%</td>
<td>58.7%</td>
</tr>
</tbody>
</table>

# Large INSERT indexes – Poor DASD Cache Hit % – High I/O response
Best use of SSD

HDD

2/3 of the READ_DISC
accommodated in 1/5 of the space

SSD

1200

GB 7800

1600

10

BP (DRAM)
Business case

• SSD order about to be finalized

• Reasonable forecast – 30% less DB2 elapsed
  • Sync_Read_Wt = 78% of elapsed time
  • Dominated by disconnect time (seek)
  • 2/3 of the disconnect time, 4ms response
  • … reduced to 0.8 ms

• Will measure post-install …
Futures ... ?

- SSD cost continues to decline
  - Cost more closely approaches HDD

- Eventually displaces HDD entirely

- DB2 elapsed time down significantly overall
  - For Sync_Read_Wt – dominated workloads, like ours

- Shorter timeline vs. “data in memory”
  - DRAM still more expensive
SSD eliminates need to REORG?

- IBM direction to reduce REORGs
  - Have some CPU, media, baby-sitting cost
  - Cause some outages (not completely 24x7)

- DB2 10 relaxes REORG Clustering Criteria for SSD
  - SYSTABLESPACESTATS.DRIVETYPE = ‘HDD’ / ‘SSD’
  - Cluster order less important, with no Seek time

- Other REORG drivers still important
  - Index disorganization (faroff leaf splits)
  - Tablespace indirect references
  - Reclaim space
  - etc.
Using Solid-State Devices to Solve a DB2 INSERT Problem

Thank you!

Terry L. Berman
DST Systems, Inc
tlberman@dstsystems.com

Acknowledgements

Albert Weaver, DST
Ramu Nalluri, DST