

5-9 November

Athens Hilton

Athens, Greece

Réunion du Guide DB2A Octobre 2008

IDUG® 2007

Europe

No Magic to Improve DB2 for z/OS
Application Performance

Marcel LEVY
Natixis

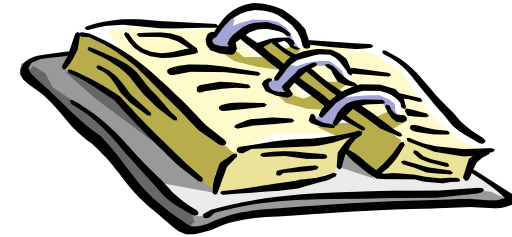


Platform: DB2 for z/OS

GoFurther



Agenda



- Story of a DB2 application migration
- Measurement of CPU, memory and I/O resources
- Understanding program logic and impact of SQL statements
- Optimizing physical design with tested techniques
- Solving DB2 performance issues with a task force

Disclaimer: Information provided without any warranty or liability and use of this information is at the sole risk of the user. The author and his company are not responsible or liable for the availability of these resources and for any content.

Acknowledgments

Special thanks for their review and help go to:

- Issac Yassin, IDUG Thread Chair
- Patrice Garault, Natixis
- Pascal Pili, Natixis
- Medhy Djema, Natixis



In Memory of Philippe Loncan (1962-2007)
DB2 DBA, a major contributor to the project



Natixis, a bank leader with a worldwide presence

Natixis is a key player in the banking sector in France and Europe, offering services centered around five core businesses: corporate and investment banking, asset management, private equity and private banking, services and receivables management. **Natixis** also consolidates a proportion of the earnings of the retail banking activities of **Caisse d'Épargne Group** and **Banque Populaire Group**.

With nearly 23,000 employees, a third of whom are based outside France, **Natixis** operates in 68 countries and supports the development of large businesses, SMEs and institutions. It generates net banking income of €7,322 million and net income (Group share) of €2,158 million.

Equally owned by **Banque Populaire Group** and **Caisse d'Épargne Group**, each of which own 34.4%, **Natixis** is listed on Paris Bourse and included in CAC Next20 index.

Story of a DB2 application migration 1/4

- Savings plan application was outsourced for years (development and hosting) and October 2006 was the planned migration date on our system
- 800 CICS programs, 1000 batch programs, 460 batch jobs
- 230 DB2 tables with 110 Gb of data and indexes
- First run with same production data and programs in June 2006 showed a longer elapsed time in our system
- Window for batch process: maximum 12 hours
 - Previous site: 6 hours 30 minutes on Friday night
 - Our site: over 9 hours, and test ran on Saturday *might be longer on weekday in our site*

Story of a DB2 application migration 2/4

- Decided to investigate the origin of the discrepancies
- I/O response time was 3 times longer in our system
 - Previous site: IBM DS8100
 - Our site: EMC 8830,
but migration planned to EMC DMX-3 within 30 days
- Reported I/O response time divided by 2
after migration from IBM ESS to DS8100 at previous site
- Alternatives:
 - Wait for new devices to resolve elapsed time issue
 - Improve performance of our site as much as possible
 - Compare both sites in order to discover the discrepancies

Story of a DB2 application migration 3/4

- Decided to go with the last idea, then improve performance in the different areas
- Decided to reduce number of I/Os in order to be less sensitive to our high I/O response time
- Decided to execute the same run in our backup system several times (same MIPS capacity and EMC disks), then in our production system with only one change at a time
- Reported improvement of each change and overall improvement (software and hardware) for every batch job

Story of a DB2 application migration 4/4

But some restrictions apply:

- New disks would be available mid-July in our system
- No application program change allowed
- No access to the previous site, only some reports or collected performance data transferred
- Decision to migrate or not would be made early September
- Only two month window to do the improvement and during summer time period
- Needed to prove we could decrease the elapsed time, even if our I/O response time was not improved.

Measurement of CPU, memory and I/O resources 1/4

- Found a job step that represented our performance problems
- Daily run, same data read, 80% time spent in DB2 I/O
- One major DB2 cursor
- No program change, no access path change
- Constant CPU and elapsed times in both sites
 - Previous site : 86 CPU seconds, 909 seconds elapsed time
 - Our site : 116 CPU seconds, 4713 seconds elapsed time
- Higher I/O response time found in our site
(4.2 ms instead of 0.8 ms in average I/O sync. time)
- Reported this I/O issue to MVS system programmers

Measurement of CPU, memory and I/O resources 2/4

- Improved just by adding a new index:
Access path changed
from non matching index scan with sequential prefetch
to matching one column with list prefetch
 - Our site: 19 CPU seconds, 1703 seconds elapsed time
- Decided to follow basic 3 R's rule (Reorg, Runstats and Rebind)
 - Our site: 16 CPU seconds, 1023 seconds elapsed time
- Modified SQL statement and existing index
 - Our site: 1 CPU second, 4 seconds elapsed time

Measurement of CPU, memory and I/O resources 3/4

- Elapsed time in previous site was “small” because of better I/O response time, but performance of program was bad
- Filtering was not sufficient to reduce the number of getpages : 2,157,000 and time spent waiting for synchronous I/O : 787 seconds in previous site
- Now, number of getpages is only 3,185 and time spent waiting for synchronous I/O : 0.5 seconds
- FYI: DB2 cursor retrieves only 34 rows from table
- Decided not to implement these changes right away
- Needed this job to show any improvement of I/O response time

Measurement of CPU, memory and I/O resources 4/4

- SEQCACH ZPARM parameter set to SEQ, as recommended in Redpaper "Disk storage access with DB2 for z/OS, REDP4187".

From DB2 Installation Guide:

SEQUENTIAL CACHE Acceptable values: ***BYPASS***, *SEQ*

Specify whether to use the sequential mode to read cached data from a 3990 controller. If you accept the default BYPASS, DB2 prefetch bypasses the cache.

If you specify SEQ, DB2 prefetch uses sequential access for read activity.

Many sites gain a performance benefit by specifying SEQCACH.

- SEQCACH ZPARM parameter had different impact depending on which disks were used.
- In QA, we had IBM disks (ESS) and it worked.
- In Production, we used EMC disks and it had no effect.

Understanding program logic

1/3

- Noticed that most jobs are sequentially dependent
 - no parallelism between applications
 - no job parallelism inside the same application
- Most CICS transactions and batch jobs are not designed to run simultaneously (commit strategy not implemented)
- Difficult to run batch jobs outside batch window
- No archive process, just a few DB2 rows deleted
- Increased the size of database by 20% each year
- Longer DB2 elapsed time (more getpages and I/Os) for jobs to retrieve the same number of rows

Understanding program logic

2/3

Identified a program with a typical "stage 3" problem

- Daily run, 85% time spent in DB2 I/O, one DB2 cursor
- Reduced number of rows retrieved, from 21 to 2.3 million by adding a DB2 predicate equivalent to COBOL verb IF
- Comparison of CPU and elapsed times
 - Previous site : 7 CPU minutes, 24 minutes elapsed time
 - Our site *before* : 8 CPU minutes, 51 minutes elapsed time
 - Our site *after* : 1.2 CPU minutes, 10 minutes elapsed time
 - Our site *3 R's* : 1.1 CPU minutes, 6.7 minutes elapsed time
- Reduced elapsed time by 87%

Understanding program logic

3/3

Detected a program with an open cursor, but no row fetched

- Perhaps the DB2 table was a sequential file a long time ago
 - open input and output files
 - open DB2 cursor
 - read parameter from linkage section
 - program will executes fetches depending on parameter value
- DB2 cursor has an order by clause and explain shows sort
- DB2 needs to sort the result of the cursor before to be able to fetch any row
- Open cursor is expensive and useless if no row fetched
- Open cursor statement was moved to another COBOL line

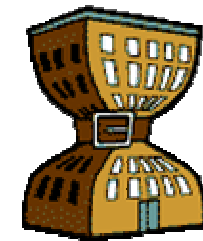
Optimizing physical design with tested techniques 1/9

- Started with basic performance checking
- Noticed same large table read several times in different jobs
- Dedicated a bufferpool to its tablespace
- Added new indexes to improve filtering
- Used DB2 Hints
 - to choose access path from previous site
 - because of change of index or sequence of tables join
 - or to favor the selection of a new index
 - not picked by DB2 Optimizer

Optimizing physical design with tested techniques 2/9

DB2 compression opportunity?

- 149 tablespaces compressed (65% of tables)
- Pagesave = 0 or negative for 54 tablespaces
- CPU wasted with no space savings
- 64Kb of dictionary pages added per partition or tablespace
- Risk of DBM1 storage shortages if no monitoring (DB2 V7)



Solution: removed DB2 compression for small tablespaces with no savings

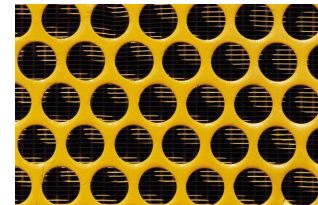
Optimizing physical design with tested techniques 3/9

TABLE	NACTIVE	NPAGES	FREEPAGE	PCTFREE	C	PAGESAVE
.../...						
TBAVCGRE	360	1	0	10	Y	-1600
TBAVCMDC	360	275	0	10	Y	76
TBAVCPPM	360	65	0	10	Y	-24
TBAVCRES	360	1	0	10	Y	-1600
TBAVCPPT	360	1	0	10	Y	-1600
TBAVCPTN	360	1	0	10	Y	-1600
TBAVCFRP	360	1	0	10	Y	-1600
TBAVCCLC	360	102	0	10	Y	17
TBAVCDFS	360	7	0	10	Y	-266
TBAVCDFE	360	122	0	10	Y	0
TBAVCDFC	360	72	0	10	Y	-22
TBAVCDDD	360	1	0	10	Y	-1600
.../...						

Optimizing physical design with tested techniques 4/9

Reducing freespace

- Implemented new freepage from 3 to 8 (standard)
- If segsize was smaller than freepage, not possible
- Two alternatives
 1. Increase segsize from 4 to 64 (standard)
 2. Reduce freepage to 0 with large pctfree
- Segsize modified with drop/create tablespace method
drawback: performance degradation w/access paths changed



Solution: reverted to former segsize and use solution 2

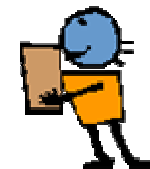
Optimizing physical design with tested techniques 5/9

TABLE	NACTIVE	NPAGES	FREEPAGE	PCTFREE	C	PAGESAVE
.../...						
TBAVCLGN	1555637	1166034	3	3	Y	55
TBAVCREL	1477034	1107116	3	6	Y	41
TBAVCSIT	1273927	954875	3	16	Y	70
TBAVCLOV	863557	575357	2	6	Y	40
TBAVCPVP	393651	393531	0	10	Y	66
TBAVCDMD	378776	283903	3	6	Y	63
TBAVCILV	374920	374868	0	10	Y	15
TBAVCOIN	286200	214512	3	11	Y	46
TBAVCMYP	278609	208822	3	2	Y	37
TBAVCCRT	246872	224355	10	10	Y	31
TBAVCCBH	170470	127765	3	11	Y	37
TBAVCPFR	146126	109518	3	3	Y	45
.../...						

Optimizing physical design with tested techniques 6/9

Reorganization of tablespaces?

- Cluster ratio less than 95%
(applies to cluster indexes of non empty tables)
- Identified 54 tablespaces
- Unsorted data was loaded with LOAD utility and no REORG utility was planned
- Runstats done, choice not to copy DB2 statistics
- Impact on DB2 access paths selection
- Better performance observed if cluster ratio is 100%



Optimizing physical design with tested techniques 7/9

NAME	TBNAME	FULLKEYCARD	CLUSTERRATIOF	NLEVELS
IAVCLGN	TBAVCLGN	48640765	.94	4
IAVCLOV	TBAVCLOV	20219341	.91	4
IAVCDMD	TBAVCDMD	12815484	.94	4
IAVCMYP	TBAVCMYP	12148570	.81	4
IAVCCRT	TBAVCCRT	11898552	.93	4
IAVCPFR	TBAVCPFR	7867490	.81	4
IAVCASR0	TBAVCASR	6682488	.78	4
IAVCPVE0	TBAVCPVE	4703619	.64	4
IAVCTRR	TBAVCTRR	1987967	.85	4
IAVCCEM	TBAVCCEM	828835	.84	3
IAVCPBF	TBAVCPBF	323919	.89	3
IAVCPAM	TBAVCPAM	302398	.84	3
IAVCPAF	TBAVCPAF	273983	.81	3
.../...				

Optimizing physical design with tested techniques 8/9

DB2 access path comparison

- Used a SQL statement to select important columns from DB2 PLAN_TABLE (found in DB2-L from Robert Blackwell)
- Unloaded sorted data from both PLAN_TABLE
- Moved spaces to erase queryno field because COBOL sources are different
- Used ISPF option SUPER CE utility 3.13 to compare
- Used SQL statements with UNION ALL to display DB2 access path from both sites with SPUFI

Optimizing physical design with tested techniques 9/9

```
***** Top of Data *****
  ISRSUPC   -   MVS/PDF FILE/LINE/WORD/BYTE/SFOR COMPARE UTILITY- ISPF FOR z/OS
NEW: OUR SITE                                OLD: PREVIOUS SITE
  LISTING OUTPUT SECTION (LINE COMPARE)
```

```
ID          SOURCE LINES
  ----+-----1-----+-----2-----+-----3-----+-----4-----+-----5-----+-----6-----+-----7-----+
I - BAVCE560..      -1-2..M...TBAVCREL.....I  IAVCREL  1..N NYNN NNNN ISS 0....
D - BAVCE560..      -1-2..N...TBAVCREL.....I  IAVCREL  3..N NNNN NNNN IS  0....
```

SIT	PROG	QURY#_B_P	M	TABLE_NAME	AT_INDXNME_MC	XNUJOGCUJOG	TSL	P	C	M
PRE	E560	01695-1-1		TBAVCMVT	I IAVCMVT	1 N NNNN NNNN	IS	S		0
PRE	E560	01695-1-2	N	TBAVCREL	I IAVCREL	3 N NNNN NNNN	IS			0
OUR	E560	02001-1-1		TBAVCMVT	I IAVCMVT	1 N NNNN NNNN	IS	S		0
OUR	E560	02001-1-2	M	TBAVCREL	I IAVCREL	1 N NYNN NNNN	IS	S		0

Solving DB2 performance issues with a task force 1/5

Why do we need a task force?



- There are three areas for improvement:
 - I/O response time on storage box
 - DB2 application performance
 - DB2 physical design



- Each team (system, development and DBA) is convinced that most of the work has to be done by the other two teams
- We need to work on three main areas simultaneously in order to get a correct elapsed time of the application before our migration date, planned in October 2006

Solving DB2 performance issues with a task force 2/5

- Task force members belong to different teams: system, development, DBA and production
- Best to share the same office in order to work together as a team
- Daily routine
 - Run procedure to restore the same data (DB2, VSAM, ...)
 - Run application and collect performance data
- All changes need to be tested before being implemented

Solving DB2 performance issues with a task force 3/5

- Extracted CPU and elapsed times from JES2 job logs
- Created reports to compare several runs of same workload
- Worked on improving top 20 programs
- Listed diagnostic, recommendations and elapsed time reduction expected
- Created a procedure to ensure DB2 hints remain in place
- Duplicated hints in a dedicated PLAN_TABLE in order to reinstall them easily if lost
- Remember, no-DB2 programs can be improved including sort steps (parameters and exits)

Solving DB2 performance issues with a task force 4/5

	PREVIOUS SITE		OUR SITE						Overall Improvement
	09/18/2006		09/20/2006		09/21/2006		09/22/2006		
Job	CPU	ELP	CPU	ELP	CPU	ELP	CPU	ELP	
Total	64	410	68	564	69	404	62	340	😊
AEPJA00	8	52	6	64	7	41	7	36	😊
AEPJ960	3	21	3	28	3	22	3	23	😞
AEPJD10	6	15	8	22	8	18	9	22	😞
AEPJD1L	4	21	5	34	6	30	6	22	😞
AEPJG80	0	6	2	26	2	18	2	14	😞
AEPJG70	1	11	2	19	2	18	2	14	😞
AEPJAE2	4	17	2	38	2	17	2	12	😊
AEPJB40	1	15	1	17	1	10	1	11	😊

Solving DB2 performance issues with a task force 5/5

- Improved efficiency after four weeks work as a team and migration was given the go-ahead
- I/O response time
 - Tested different configurations and microcode changes
 - Observed heavier I/O workload on our site
 - Dedicated former EMC 8830 to this application as a workaround. Result: better I/O response time
- DB2 application performance
 - Development team modified several programs because of intensive testing (daily routine)
- DB2 physical design
 - Reduced size of database (freespace) and DB2 I/Os requested

No Magic To improve DB2 for z/OS Application Performance

Marcel LEVY

Natixis

marcel.levy@natixis.fr

