IBM®DB2® to CA IDMS™ Conversion Project

Grant Epstein
John Gertner
HP Enterprise Services

CA IDMS™ Technical Conference

Framingham MA
December 2-5, 2014

Agenda

- The Thought Process
- Environment
- DBA Actions
- Programmer Actions
- Issues Encountered
- Results
- Considerations
- Summary
- Q & A

© 2014 CA. ALL RIGHTS RESERVED.

The Thought Process

- Cost savings
  - Save on DB2 license fees
  - Free up DB2 DBA's for other accounts
  - No need to upgrade or maintain DB2 system software
  - Simplified disaster recovery
- Degree of difficulty/complexity
  - Small DB2 batch only application
  - About 50 programs
  - Achievable with a small team
  - No additional functionality requirements ("stabilised")
  - Completed in 3 months (April – June 2014)

© 2014 CA. ALL RIGHTS RESERVED.

Environment

- 1 LPAR
- z/OS 1.11, DB2 v9 and CA IDMS r17
- Multiple applications running under CA IDMS, DB2 and Objectstar
- 133GB DB2 database with 26 tables
- Largest table has 28 million rows
- 50 DB2 batch COBOL programs

© 2014 CA. ALL RIGHTS RESERVED.

DBA Actions - Slide 1 of 3

- Database conversion:
  - Create the application catalog
  - Define this to the dbtable
  - Populate the application catalog with the system tables and views
    - R171.DISTSRC(TABLEDDL)
    - R171.DISTSRC(VIEWDDL)

© 2014 CA. ALL RIGHTS RESERVED.

DBA Actions - Slide 2 of 3

- Define the DDL to the application catalog
  - Create SQL database
    - Create the Tables
    - Create Indexes
    - Create Keys
  - DROP the “DEFAULT INDEX” from each table
    - This is indexed on DBKEY and not required since we defined several UNIQUE indexes

© 2014 CA. ALL RIGHTS RESERVED.

DBA Actions – Slide 3 of 3

- Extract data using DB2 UNLOAD utility
- Populate data using CA IDMS LOAD utility
- Run UPDATE STATISTICS
  - Used by the precompiler as input to the optimizer to create the AM (Access Modules)
- Convert DB2 utility statements to CA IDMS / other utilities
  - COPY TABLESPACE to DFDSS Backup
  - RUNSTATS to UPDATE STATISTICS
- Deactivate obsolete DB2 jobs & region

© 2014 CA. ALL RIGHTS RESERVED.

Programmer Actions

- Perform program code changes
- Make JCL Changes
- Run tests of the current DB2 code
- Run the same tests of the new CA IDMS SQL code
- Compare the two sets of test results to show that there are no differences
  - Use production size data to ensure that there are no performance issues

© 2014 CA. ALL RIGHTS RESERVED.

Code Changes – Slide 1 of 4

- Some values returned in SQLSTATE and SQLCODE are changed under CA IDMS SQL
  - CA IDMS SQL has a small list of SQLCODE values, therefore SQLSTATE should be used to identify the actual error
- Also check code for any hard coded compares against SQLSTATE and SQLCODE, and alter as per next slide, changes highlighted:

© 2014 CA. ALL RIGHTS RESERVED.

Code Changes – Slide 2 of 4

Description	IDMS/DB2	SQLCODE	SQLSTATE
OK	IDMS	0	00000
	DB2	0	00000
End set/No Record	IDMS	+100	02000
	DB2	+100	02000
Multi	IDMS	-4	21000
	DB2	-811	21000
Duplicate	IDMS	-4	23501
	DB2	-803	23505
Invalid Date	IDMS	-4	22007
	DB2	-180/-181	22007
Deadlock	IDMS	-5	40001
	DB2	-911	40001

© 2014 CA. ALL RIGHTS RESERVED.

Code Changes – Slide 3 of 4

- For update programs, ensure programs have an end of program COMMIT
  - DB2 appears to have an implied end of program commit
  - CA IDMS SQL does not
- COBOL copybooks for SQL declarations needed an additional EXEC SQL BEGIN DECLARE / EXEC SQL END DECLARE
- Hard coded host variables in the working-storage section also needed an additional EXEC SQL BEGIN DECLARE / EXEC SQL END DECLARE
- If copybooks are stored outside the IDD environment, relocate Code and Data Copybooks into the CA IDMS IDD environment, and use 'COPY IDMS xxxxxxx', where xxxxxxx = actual IDD copybook element

© 2014 CA. ALL RIGHTS RESERVED.

Code Changes – Slide 4 of 4

- Include within the program code

```
IDMS-CONTROL SECTION.
PROTOCOL.    MODE IS BATCH-AUTOSTATUS DEBUG
             IDMS-RECORDS MANUAL.
```

- Replace the Cursor declaration 'HOLD', with COMMIT CONTINUE
- If a program uses the SELECT BETWEEN option which performs an area sweep on an entire table of rows, for large input files this can cause performance issues, even if just for a 50,000 row table
  - Change the code to load the rows into an internal COBOL table and scan it
  - This should result in a huge performance improvement for the production runs
- If there are any DB2 statements using data names that include a '-' (hyphen), they should be changed to a '_' (underscore)

© 2014 CA. ALL RIGHTS RESERVED.

Compiler Changes

Changes were made to the compile process

Precompiler

- PRECOMP EXEC PGM=IDMSDMLC,REGION=0M

Access Module

- ACCMOD EXEC PGM=IDMSBCF,REGION=1024K
  - Where you need to link in any user called modules

Compile step

- COMPILE EXEC PGM=IGYCRCTL,REGION=240K

Link Program Module

- LINK EXEC PGM=IEWL,REGION=300K

© 2014 CA. ALL RIGHTS RESERVED.

JCL Changes

- Change 'EXEC PGM=IKJEFT1B' to 'EXEC PGM=xxxxxxxx', where xxxxxxxx is the actual program name
- Include, 'INCLUDE MEMBER=$IDMSCV'
  - For our site this points to a set of IDMS LOADLIBs
- Other JCL changes will depend on the site standards

© 2014 CA. ALL RIGHTS RESERVED.

Issues Encountered – Slide 1 of 5

DBA Issues reported to CA:

- CREATING APP SYSTEM TABLES
  - We had forgotten to run the TABLEDDL and VIEWDDL
  - Make sure you do this before defining your application DDL

- LOAD SQL TABLE – SLOW
  - Trying to load 10 million rows of data
  - Job ran for over 10 hours and had performed 85 million EXCPs
  - Improved performance for LOAD utility by
    - Increasing the page size from 4K to 32K and using PREFETCH
    - Also increased FILE_BUFFER statements to 500 each
    - LPAR CPU allocation increased

© 2014 CA. ALL RIGHTS RESERVED.

Issues Encountered – Slide 2 of 5

- LOAD NULL IF CLARIFICATION
  - Received error DB005505 C-4M322: Datetime character constant format because data had X'40' values and the column defined as TIMESTAMP
  - In CA IDMS the NULL indicator follows the data
    - The unloaded data had the NULL indicator before the data
    - The data field contains binary 0s not blanks
    - Used TERM_TXN_TIMESTAMP POSITION 00124 TIMESTAMP NULL IF X'40'

Example

DB2 LOAD -

```
      , "TERM_TXN_TIMESTAMP"
        POSITION(  00124:00149) TIMESTAMP EXTERNAL
                                NULLIF(00123)=X'FF'
```

IDMS LOAD -

```
      , TERM_TXN_TIMESTAMP
        POSITION    00124       TIMESTAMP
                                NULL IF X'40'
```


© 2014 CA. ALL RIGHTS RESERVED.

Issues Encountered – Slide 3 of 5

Application Issues reported to CA:

- SQL PERFORMANCE
  - We have a batch SQL program that is performing an open cursor and is taking over 34 seconds to complete;
  - Several problems: Buffers were too small, PREFETCH was on. We fixed this by adding additional indexes and this improved the performance.
- SQL INSERT - POOR PERFORMANCE
  - SQL INSERTs slow
  - Resize the indexes: Make sure there are at least 4 SR8s per page. This reduced load time from 6.5 hours to 2.25, this allowed the INSERTs to run in seconds rather than minutes.

© 2014 CA. ALL RIGHTS RESERVED.

Issues Encountered – Slide 4 of 5

- DC203005 - DUPLICATE VIOLATION
  - These messages were being written to the CA IDMS LOG (around 200 a day), no messages written when under DB2
  - We could enhance the program to check for duplicates before we attempt to INSERT
- DECLARE CURSOR WITH HOLD
  - Converting DB2 COBOL programs to CA IDMS SQL COBOL has a declare cursor with hold
    - EXEC SQL DECLARE CUR_PEND CURSOR WITH,HOLD,FOR SELECT SERVICE_NUMBER,
  - The equivalent in CA IDMS, to keep the cursor open after a commit, is to use COMMIT CONTINUE

© 2014 CA. ALL RIGHTS RESERVED.

Issues Encountered – Slide 5 of 5

- DSNTIAR EQUIVALENT FOR CA IDMS
  - DSNITIAR is the assembler subroutine that helps you to obtain a formatted form of the SQLCA and a text message that is based on the SQLCODE field of the SQLCA
  - In CA IDMS, an assembler program can call IDMSIN01 with a GETMSG function
    - It retrieves SQL error messages and places them in a user buffer that is displayed to the user
    - Documented in the IDMS Callable Services manual

© 2014 CA. ALL RIGHTS RESERVED.

Results – Slide 1 of 2

- Resource impacts
  - CPU usage was nearly the same
  - DASD usage went up by approximately 20% since we tried to ensure the new CA IDMS databases were sitting around 70% utilization
  - Tape usage was approximately the same. A decrease as a result of not needing to backup DB2 system files / runtime libraries, and an increase associated with the larger database allocation.
- Project costs
  - 2 man project – 3 months
  - Temporary additional DASD and CPU to load full production data
  - Costs were fairly low

© 2014 CA. ALL RIGHTS RESERVED.

Results – Slide 2 of 2

- The modified application performed the same as it had under DB2
  - Batch times were comparable and some jobs ran slightly quicker and some slightly slower
    - Performance improved with additional indexes that performed Leaf Scans
    - And large buffers - 10,000 pages
- The customer and the account were delighted with the conversion and associated cost savings.

© 2014 CA. ALL RIGHTS RESERVED.

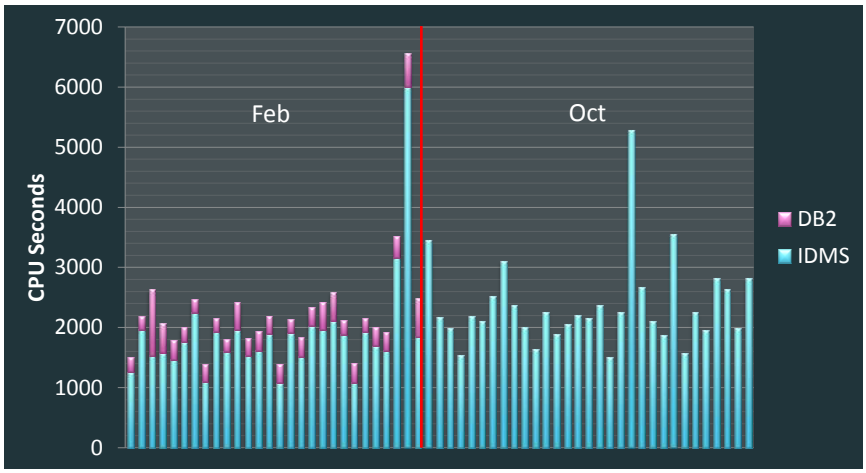
DB2 and IDMS CPU Times – Slide 1 of 2

- Before – average CPU time was 2257 seconds per day
- After – average CPU time was 2363 seconds per day

An increase of approximately 5% between the 2 months sampled.

© 2014 CA. ALL RIGHTS RESERVED.

DB2 and CA IDMS CPU Times – Slide 2 of 2



© 2014 CA. ALL RIGHTS RESERVED.

DDL Considerations – Slide 1 of 2

Provided by CA:

- CA IDMS does not support the PRIMARY KEY clause in the CREATE TABLE statement
  - Primary keys are defined using CREATE UNIQUE CALC or CREATE UNIQUE INDEX statements instead.
    - We used both of these statements in our conversion
- CA IDMS does not support table and constraint names >18 chars
- VARCHAR column lengths are limited by page size
- Primary keys are limited to 256 characters

© 2014 CA. ALL RIGHTS RESERVED.

DDL Considerations – Slide 2 of 2

- CA IDMS uses ADD/DROP CONSTRAINT instead of ALTER TABLE for adding and deleting constraints
- CA IDMS does not support the UNIQUE clause in a column definition
  - A column which doesn't permit duplicates must be specified with CREATE UNIQUE CALC or CREATE UNIQUE INDEX
- CA IDMS does not support a FOREIGN KEY clause in a constraint definition
- CA IDMS doesn't support column-level check constraints
  - These can be added as table-level constraints, separated by "AND" operators

© 2014 CA. ALL RIGHTS RESERVED.

Programming Considerations

Provided by CA:

- There is no pre-compiler for CA IDMS Assembler or "C" programs
- SQLCODE is less-detailed in CA IDMS
  - CA IDMS programs are expected to query SQLSTATE (which is standardized, unlike SQLCODE) to evaluate error conditions
- There are some differences in dynamic SQL, like cursor allocation

© 2014 CA. ALL RIGHTS RESERVED.

Functions and Procedure Considerations

Provided by CA:

- DB2 functions and procedures must be restructured into their CA IDMS equivalents
- DB2 table functions must be implemented as table procedures in CA IDMS

© 2014 CA. ALL RIGHTS RESERVED.

Summary – Slide 1 of 3

- Ensure that the DB2 database structure would be compatible with CA IDMS
  - As we previously discussed
- Perform a Proof of Concept
- Additional DASD for the project
  - Needed to support both the CA IDMS and DB2 version until the conversion was completed
  - Sort-work for the LOAD
    - We used 12,000 cylinders for our largest table
  - Create a new Application Catalog for each CA IDMS environment

© 2014 CA. ALL RIGHTS RESERVED.

Summary – Slide 2 of 3

- LPAR CPU needed to be increased temporarily to allow the LOADS to complete in the allotted change window
  - We had only a few hours to complete the cutover in production and the Loads were taking too long under the existing CPU allocation
- Prepare yourself for someone asking why you're moving from DB2 to CA IDMS
  - The mainframe was seen as "legacy" and so there wasn't much differentiation between CA IDMS and DB2
  - You will know your own environment and decision makers

© 2014 CA. ALL RIGHTS RESERVED.

Summary – Slide 3 of 3

Performance:

- Trying to understand why some programs were taking longer to execute
  - To really identify potential performance issues you need a production sized database
- Running access path reports
  - Running DB2 and CA IDMS EXPLAINS and comparing the access paths
- Creating new indexes to improve performance
  - Leaf scan indicator, when ACMODE is I
  - This indicates whether data is retrieved by sequential access to index leaf pages

© 2014 CA. ALL RIGHTS RESERVED.

And finally…

Thank you to CA Support for all the help they provided with the issues we encountered and their excellent advice

© 2014 CA. ALL RIGHTS RESERVED.

Online Session Evaluation

Please provide your feedback about this session: D9

On the CA Communities web site: http://communities.ca.com

More details in your conference bag

Questions and Answers