Db2 Hot topics from Progressive Insurance

Dustin Ratliff & Bob Vargo

19 September 2023

Agenda

- Db2 AI install challenges
- Db2 AI usage challenges
- Db2 driver upgrade issues
- Db2 .Net core driver issues
- Windows 11/Kerberos issues
- IBM Replication center issues

Db2 AI at Progressive

- Db2 AI was brought in for a POC effort
  - We wanted to take a look at the install portion
  - We wanted to connect Db2 AI to our sandbox (1 way) and QA (2 way) data sharing groups to take a look at the features and benefits
  - Our intention was to target our QA data sharing members as target members and not fully install the product on those LPARs
    - » The LPAR where our sandbox member runs is smaller
  - It was never intended to go to PROD with this effort

Db2 AI install - SMPE

- As a Db2 SYSPROG I handled the Db2 AI install portion while our z/OS systems programmer handled the rest (WML, spark, etc)
- The initial SMPE install of Db2 AI and related products went smoothly
- A few notes on the initial SMPE install:
  - Ensure that JAVA_HOME is set as your install directory within your jobs
  - To enable Db2ZAI 1.5 to run with z/OS, you must set up Dynamic Enablement. For instructions, see the Dynamic Enablement section of the program directory.
  - We setup different SMPE global zones for Db2 AI itself and the other products (WML, spark, etc)

Db2 AI install and challenges – Post SMPE

- Automation of the Db2 AI STCs
  - These tasks are OMVS tasks and do not cut messages to the syslog upon startup or shutdown
  - The Db2 AI tasks spin up multiples and have numbers at the end of some of them which are not the same every time:
    DBAIND3 (the end number here will be 1-9)
    DBAIND
    DBAIND
  - For these tasks the one with the number on it is the main task.
  - If any non numbered task fails, it is suppose to self restart

Db2 AI install and challenges – Post SMPE

- Automation of the Db2 AI STCs
  - The solution we ended up implementing was to set a timer and every 5 mins “watch” the Db2 AI STC with the number at the end
    - » Ops had to wait until the product was started to determine which one this was as the number changed each time from 1-9
  - At IPL time, this timer would decrease to every 1 min so that if the tasks did not come down we would not hang system shutdown

Db2 AI install and challenges – Post SMPE

- The STCs required symbolic links due to the 100 character limit on the PARM value in the JCL
  - Within the startup JCL for the liberty server and Db2 AI STCs there are parm values which in JCL have a 100 character limit
  - We reached this limit and did symbolic links within OMVS to get around it: /install/abcdefgh/ijklmnop/qrs/tuv/wxyz became /install/db2ai

Db2 AI install and challenges – Post SMPE

- There were so many ports required for the product
  - Within the install doc it asks you to reserve 27 network ports for the various pieces and parts of the product
  - These are all aside from your normal Db2 ports you are already using

    List of products needing all of these ports:

    z/OS Spark master, z/OS Spark master REST API, z/OS Spark master UI, z/OS Spark worker, z/OS Spark worker UI, z/OS Spark executor, z/OS Spark driver, z/OS Spark driver block manager, Spark-integration service, Scoring service, WMLz base UI service, WMLz base core services, Configuration tool service, Db2ZAI user interface, Db2ZAI Liberty server

Db2 AI install and challenges – Post SMPE

- Figure out the TCP thing which bob had to fix prior to getting passtickets to work.

Db2 AI usage challenges

- Once we got the product up and running, it filled up the /tmp directory inside of OMVS
  - What happened was once we started the AI and liberty server tasks, It started learning about the target Db2s we connected it to.
  - As it learns, it stores data in the /tmp directory in OMVS
  - In order to correct this, we had to set the TMPDIR evn variable to specify where AI is to put the temp data
    ex: export TMPDIR=/newTemDir
  - Or if you started it from an STC proc you'd set the TMPDIR env under the STDENV DD card

Db2 AI usage challenges

- When we tried to kick off a system assessment liberty server consumed nearly all of the AUX storage on the LPAR where it was running.
  - The system assessment which I ran was looking back at a weeks worth of data
  - The IBM documentation states that 25GB additional storage is needed while system assessments and training is executing
    - » We have this on the LPAR but not a lot more

Db2 AI usage challenges

- When we tried to kick off a system assessment liberty server consumed nearly all of the AUX storage on the LPAR where it was running.
  - When I tried to stop the liberty server there was not enough AUX storage to spin up the address space to do so, I had to hard cancel it to free the storage
  - We did not have to IPL to get out of this
  - We did have another crash of this LPAR later in this POC which we suspected Db2 AI to be involved with as well

Db2 AI usage challenges

- Throughout the POC, due to previously mentioned challenges we were not able to run a full system assessment
  - Mainly between the filling of the TMP directories and the AUX storage shortages each time we tried the system assessment failed and nearly crashed the systems

Db2 AI usage challenges

- When it first was connected to a target Db2, it kicked off its learning and a system assessment which did get us partial information which we could look at
  - One thing we noticed when looking at the DCC capabilities was that the graphs showing clients with WLB enabled seemed incorrect
  - We were able to see the IPs listed and navigate to those clients and show within their db2 cfg files they were running with enableWLB=true
    - » We suspect that this feature is using the ATT field within a –DIS LOCATION command which sometimes specifies WLB indicating the client is using a sysplex WLB connection
    - » We saw by issuing the DIS LOCATION command that it also did not always seem correct with this information

Db2 AI Positives

- They have decoupled the need to WML to be installed with Db2AI
- The DCC would be nice to learn more about the DDF traffic including connection floods and possible areas where we need to tweak things like WLB
- Profiling recommendations would be very helpful and interesting
- We believe that the SQL optimizations could be very helpful (YMMV, if watched)
- We believe that the direction and vison for this product is a good one and are still interested in it, however with our experience it was not something we wanted to implement in PROD at this time.

Questions?

Agenda – Part 2

- Fast Traversal Block (FTBs) use at Progressive
- Batch Generation of V13 Migration Jobs
- Using Profiles to Control DDF workload

Fast Traversal Blocks – Background

- Fast index traversal (aka FTB) is a process that can improve the performance of random index access.

- FTBs use memory outside of the Db2 buffer pools

Controlling FTBs

- SYSIBM.SYSINDEXCONTROL
  - This table can be used in conjunction INDEX_MEMORY_CONTROL to enable, disable or force FTB usage for specific indexes
- ZPARMs
  - INDEX_MEMORY_CONTROL
    - DISABLE
    - AUTO
    - A Storage Amount (meg)
    - SELECTED (can be AUTO or a Storage Amount in meg)
  - FTB_NON_UNIQUE_INDEX (Yes or No)

Our Implementation

- We do not use SYSIBM.SYSINDEXCONTROL
- INDEX_MEMORY_CONTROL is set to a storage amount (for example: 512)
  - In production we used a small amount to start, much less than 20% of the total buffer pool size
- We have FTB_NON_UNIQUE_INDEX=NO for now

Monitoring FTBs

- Fields in IFCID 2
- -DIS STATS
  - ITC: INDEXTRAVERSECOUNT
  - IMU: INDEXMEMORYUSAGE
- IFCID 389
- IFCID 477

FTB info in IFCID 2 – mapped by DSNDQIST

- QISTTRAVMIN – Internal value – it's always 1000 at the moment. It represents the minimum threshold of index traversals

- QISTFTBCANT - Total number of indexes which meet FTB criteria
  - It's actually the number of **OPEN INDEX PARTS** that which meet FTB criteria

FTB info in IFCID 2

- QISTFTBCAN - Total number of **OPEN INDEX PARTS** which meet FTB criteria and the traverse count is above the threshold (QISTTRAVMIN = 1000)
  - Re-evaluated every two minutes
  - **Question:** why aren't they all in use as FTBs ?
- QISTFTBSIZE - total memory allocation for all FTBs for this member (In Meg)
  - This may be less than the potential

FTB info in IFCID 2

- QISTFTBNUMP - Number of **Index Parts** for which FTB existed in the previous run of in-memory optimization (Prior two minute interval)

- QISTFTBNUMC - Number of **Index Parts** for which FTB exists in the current run of in-memory optimization (for the current two minute interval)

-DIS STATS(IMU) LIMIT(*)

- Displays the index parts, in descending order by memory usage, that are currently active.

- V12 example:

```
DSNT783I
DBID  PSID  DBNAME   IX-SPACE   LVL  PART SIZE(KB)
-----  -----  --------        --------        ---  ----  --------
00418 00297 SAMPDB1 INDEX123  004  0001  00045152
00435 00016 SAMPDB2 INDEX345  004  0001  00024064
00267 00100 SAMPDB3 INDEX567  004  0001  00012683
```

-DIS STATS(ITC) LIMIT(*)

- With LIMIT(*) all index parts that are eligible will be displayed (QISTFTBCANT)
- The V12 display is descending by Traversal Count
  - Recent maintenance may change this – V13 changes have been retrofitted to V12
- The V13 display is descending by FTB Factor

-DIS STATS(ITC)

- The display can be qualified by DBNAME, SPACE & PART
- V12 display:

```
DSNT830I
DBID  PSID  DBNAME    IX-SPACE     LVL      PART     TRAV. COUNT
-----    -----    --------       --------          ---       ----       ----------
00406 00094 SAMPDB2 INDEX001    003      0001      0000452354
```

- V13 display:

```
DSNT830I
DBID  PSID    DBNAME   IX-SPACE LVL PART TRAV.  COUNT FTB FACTOR
 -----    -----    --------       --------        ---   ----     ----------          ----------
00444 00019 SAMPDB5 INDEX002 003 0001   0000000000    0000000000
```

IFCID 389

- This is the same data from the –DIS STATS(IMU) LIMIT(*) command.
- We send STATS(*) to SMF and this IFCID is covered. The IFCID cuts every two minutes – each time the FTBs are re-evaluated
- V13 now includes the FTB Factor for each index part

IFCID 477

- This IFCID tracks the create/free of FTBs
- Not externalized with STAT(*)
- Cuts on the two minute interval

Our results

- We have a large number of index parts that qualify based on traversal count (> 1000)
- Memory utilization is between 30 – 40% of our specified amount
- Getpage decrease is noticeable – CPU decrease has been more difficult to discern
- Overhead hasn't increased

It would be nice if . . .

- Index parts that have a high traversal count combined with a low FTB factor could be more easily tracked

Batch Generation of V13 Migration Jobs

- Supplied by V13 APAR PH52482 / PTF UI91497

- COMMENTS:

This PTF adds new parts DSNTIDOM, DSNTIDON, DSNTIDOA, and DSNTIJBC in the prefix.SDSNSAMP target library, adds a new program DSNTIFMT in the prefix.SDSNLOAD target library, and generates a new Db2 installation CLIST, which can run in the background and enables users to generate tailored Db2 migration or function level activation jobs.

First Step

- Use DSNTXAZP to generate TIDXA members (Db2 Installation Data) at V12 for all subsystems
- Used as input to the install/migrate clist
- Not needed if you keep these up to date

DSNTIJBC

- The job has three steps:
  - Run DSNTIFMT to reformat the install clist (DSNTINST) to run in batch (DSNTINSB)
  - IEBGENER to print DSNTINSB to SYSOUT
  - Invoke DSNTINSB with ISPF batch

DSNTIDOM

- Parms used to generate migration jobs for a standalone Db2 or for the first member of a data sharing group

BATCH_MODE=YES
USE_ZOSMF_WORKFLOW=NO
INSTALL_TYPE=MIGRATE
MIGRATE_INPUT_DATA_SET=< V12 TIDXA for this subsystem >
DATA_SHARING=YES
MIGRATE_FIRST_GROUP_MEMBER=YES
DB2_SMPE_LIBRARY_NAME_PREFIX=< Prefix of V13 SMPE datasets >
DB2_SMPE_LIBRARY_NAME_SUFFIX=
INSTALL_DATA_SET_PREFIX=< Prefix for generated datasets >
INSTALL_DATA_SET_SUFFIX=< SSID >
DEFAULT_PARAMETER_INPUT_MEMBER=DSNTIDXA < V13 Shipped version >
PARAMETER_OUTPUT_MEMBER=< Generated V13 TIDXA for this member >
TARGET_FUNCTION_LEVEL=
CONSOLE_NAME=

DSNTIDON

- Parms used to generate migration jobs for additional member(s) of a data sharing group

```
BATCH_MODE=YES
USE_ZOSMF_WORKFLOW=NO
INSTALL_TYPE=MIGRATE
MIGRATE_INPUT_DATA_SET=< V12 TIDXA for this subsystem >
DATA_SHARING=YES
MIGRATE_FIRST_GROUP_MEMBER=NO
DB2_SMPE_LIBRARY_NAME_PREFIX=< Prefix of V13 SMPE datasets >
DB2_SMPE_LIBRARY_NAME_SUFFIX=
INSTALL_DATA_SET_PREFIX=< Prefix for generated datasets >
* The DEFAULT_PARAMETER_INPUT_MEMBER was generated
*   by the first job. It's the V13 DSNTIDXA that was output from that job
DEFAULT_PARAMETER_INPUT_MEMBER=<V13 DSNTIDXA from first member>
PARAMETER_OUTPUT_MEMBER=< Generated V13 TIDXA for this member >
CONSOLE_NAME=
```

DSNTIDOA

- Parms for activating a Db2 function level

BATCH_MODE=YES
USE_ZOSMF_WORKFLOW=NO
INSTALL_TYPE=ACTIVATE
DB2_SMPE_LIBRARY_NAME_PREFIX= =< Prefix of V13 SMPE datasets >
DB2_SMPE_LIBRARY_NAME_SUFFIX=
INSTALL_DATA_SET_PREFIX=<Careful here – read the doc>
DEFAULT_PARAMETER_INPUT_MEMBER==<Valid V13 DSNTIDXA >
PARAMETER_OUTPUT_MEMBER=<New TIDXA for this member>
TARGET_FUNCTION_LEVEL=V13R1M5xx

Our Implementation

- Generate V12 TIDXA members for all subsystems. We do not keep these up to date.
- Use a homegrown variable substitution utility to generate TIDOM & TIDON members and batch jobs to run DSNTINSB.
- The original migration jobs are tailored and then cloned for use for subsequent data sharing group migrations.

Controlling DDF with Profiling

- We have a number of profiles that have been used to try to control connection flooding
- At times these profiles have worked when –STOP DDF MODE(FORCE) on all members has failed to control the connection flood

Controlling DDF with Profiling

- The Hammer: Use MONITOR ALL CONNECTIONS with EXCEPTION_DIAGLEVEL2 for Location 0.0.0.0. The number of allowable connections is set to a very small value.

- The problem connections bleed off and this gives us time to shut down the offending servers.

Controlling DDF with Profiling

- Smaller Hammer: Use MONITOR ALL CONNECTIONS with EXCEPTION_DIAGLEVEL2 for a specific location. The number of allowable connections is set to a very small value but it only applies to one location. Limiting threads by AUTHID also works.

- This also buys time to shut down the problem server.

Controlling DDF with Profiling

- These profiles are kept in the profile tables with PROFILE_ENABLED set to ‘N’ so that they can be activated when needed.

- We also have samples that can be quickly changed when new locations cause issues.

Controlling DDF with Profiling

- Our monitoring checks for connection flooding every minute on every production subsystem. We identify flooding very quickly.

Controlling DDF with Profiling

- Question: Does this work ?
  - It depends
- Question: would Db2 AI work better ?
  - Again – it depends

Questions ?

Appendix

Running DSNTIJBC

- We condensed all of the steps into one proc

```
//STEP02   EXEC TIJBC
//STEP01.SYSIN    DD  *
//BATISPF.SYSTSIN   DD  *
  ISPSTART CMD(%DSNTINSB +
    OVERPARM(<Parm.Library>(<Parm_mem>))
) BREDIMAX(1)
```

Parm_mem is a specific TIDOM, TIDON or TIDOA member

Running DSNTIJBC with VUE

- The clist invocation has to change for VUE

```
//STEP02   EXEC TIJBC
//STEP01.SYSIN    DD  *
//BATISPF.SYSTSIN  DD  *
  ISPSTART CMD(%DSNTINSB +
    OVERPARM(<Parm.Library>(<Parm_mem>)) +
    OTCLPARM(Parm.Library(DSNTIDVU)) +
) BREDIMAX(1)
```

Parm_mem is a specific TIDOM, TIDON or TIDOA member – same as before

DSNTIDVU must have these settings:

```
OTC_LICENSE_USAGE=YES
LICENSE_TERMS_ACCEPTED=YES
```

Tracing for DSNTINSB

- You can pass these trace parms to the invocation of DSNTINSB:
  - CONTROL(L) - LIST
  - CONTROL(C) – CONLIST
  - CONTROL(S) - SYMLIST

```
 ISPSTART CMD(%DSNTINSB CONTROL(S) +
   OVERPARM(<Parm.Library>(<Parm_mem>)) +
   OTCLPARM(Parm.Library(DSNTIDVU)) +
) BREDIMAX(1)
```

Debugging for DSNTINSB

- You can also run the format program (DSNTIFMT) and save a copy of DSNTINSB. This can come in handy until you get the parms set properly. You can use this to add additional tracing (WRITE statements) if need be.

- The error messages aren't always informative.

Parms for DSNTIDOM,ON & OA

- Each of these members has a set of required parms and a set of optional parms. You should carefully review the descriptions for all of the parms.

- The parms in this presentation were the required parms that worked at our shop. We also use some of the optional parms as well.