WHAT YOU MUST KNOW ABOUT DISTRIBUTED ACCESS TO DB2 FOR Z/OS – PART 1
Cristian Molaro, Belgium
Independent consultant
IBM GOLD Consultant
IBM Champion IM
cristian@molaro.be
@cristianmolaro
About the speaker

➔ Cristian Molaro, MConsulting Bvba, Belgium
  – Independent DB2 specialist and an IBM Gold Consultant
  – Recognized by IBM as "TOP" EMEA Consultant in 2011 / 2013
  – Co-author of 9 IBM Redbooks related to DB2. Holder of the merit badge “Platinum IBM Redbook Author”
  – 12 books published about DB2 for z/OS and DB2 for LUW

➔ Reachable at
  – cristian@molaro.be
  – be.linkedin.com/in/cristianmolaro/

© 2015
DISCLAIMER

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

Objective
- To review topics that you **must** know about distributed access to DB2 for z/OS

Contents
- Connectivity topics
- Accounting and SMF
- High performance DBATs
- And there is more
- In Part 2:
  - Data Sharing considerations
  - Specialty engines
  - What is new in DB2 11
  - Problem determination
CONNECTIVITY TOPICS
Distributed access to DB2

Configuration examples
Distributed Relational Database Architecture levels

→ Communication will be done using the lowest DRDA level supported by the Client / Server

→ Working with down-level clients?
  – An old client will work but probably with a subset of the DRDA capabilities of the DB2 server
  – Clients and servers are supported independently

→ **BUT:** feedback from IBM DDF Level 2 Support area shows:
  – Typical problem: distribution protocol errors or errors with certain DDM code points
  – Special register settings not taking effect after connection reuse
  – Many (sometimes undetermined) problems solved after updating clients

**IMPORTANT:** Keep clients up to date
Database access threads processing modes

 ➔ **ACTIVE MODE:** A DBAT is always active from initial creation to termination

 ➔ **INACTIVE MODE:** A DBAT that is not currently processing a unit of work is called a pooled thread, and it is disconnected
  – When a DBAT in INACTIVE MODE is active, it processes requests from client connections within units of work
  – When a DBAT is pooled, it waits for the next request from a client to start a new unit of work

 ➔ **CONDBAT:** Max. # of distributed connections into DB2 system
  – Includes inactive and active connections, may be large
  – DB2 queues DBAT requests to become active up to CONDBAT

 ➔ **MAXDBAT:** Max # database access threads (DBATs) that can be active concurrently
  – In many installations, max. value determined by available storage in DBM1 → check IFCID 225
Use INACTIVE threads

➔ Enable inactive thread support
  – CMSTAT=INACTIVE
  – Allows DB2 for z/OS pooling:
    • Reduction in CPU utilization
    • Reduction in Memory utilization

➔ To allow DDF threads to become INACTIVE
  – Avoid holding resources
    • WITH HOLD cursors not closed
    • DTT not dropped
    • Application using packages bound using KEEPDYNAMIC

IMPORTANT: Resources held across a COMMIT would prevent the connection and associated DB2 thread from being POOLED
DIS DDF command

```
DSNL080I -DB2P DSNLTDDEF DISPLAY DDF REPORT FOLLOWS:
DSNL081I STATUS=STARTD
DSNL082I LOCATION LUNAME GENERICLU
DSNL083I DB2P DB2P.LU1 NONE
DSNL084I TCPPORT=5136 SECPORT=5137 RESPORT=5138 IPNAME=-NONE
DSNL085I IPADDR::192.168.1.1
DSNL086I SQL DOMAIN=WWW.HELLOWORLD.BE
DSNL090I DT=I CONDBAT=1000 MDBAT=200
DSNL091I ADBAT=2 QUEDBAT=0 INADBAT=0 CONQUED=0
DSNL092I DSCDBAT=0 INACONN=0
DSNL099I DSNLTDDEF DISPLAY DDF REPORT COMPLETE
***
```

- **DT=I**  --> DDF configured with INACTIVE threads
- **CONDBAT**  --> MAX REMOTE CONNECTED
- **MDBAT**  --> MAX REMOTE ACTIVE
- **ADBAT**  --> Current # of DBATs, active and disconnected
- **QUEDBAT**  --> Count # times MDBAT was reached, only reset at restart
- **INADBAT**  --> Current # of inactive DBATs, DISPLAY THREAD TYPE(INACTIVE)
- **CONQUED**  --> Current # of queued connections
- **DSCDBAT**  --> Current # of disconnected DBATs= DBAT pool threads
- **INACONN**  --> Current # of inactive connections
IMPORTANT: DB2 Connect licensing is required in ALL configurations
Some DB2 Connect reserved functionalities

**IMPORTANT:** There is no mechanism available to DDF or WLM to classify a workload **BEFORE** connection: critical and low priority workloads compete for DBATs

→ **DB2 Connect:**
  - Provides gateway, connection concentration and a larger scope for WLB and Pooling
  - Simplification of upgrades and maintenance
DB2 Connect and Hipersockets

- DB2 Connect + zVM + zLinux
- Probably the best option for a DB2 Connect server
- Get availability advantages of System z at IFL price
- Hipersockets support
- Promotes server consolidation: reduces Data Center costs

- Even better?
  - Application server in zLinux
db2 ping

Tests the network response time of the underlying connectivity between a client and a database server

Can simulate ≠ packages sizes (bytes) for REQUEST and RESPONSE

Example:

C:\..\SQLLIB\BIN>db2 ping PRODDB request 100 response 32000 10 times

Elapsed time: 4256 microseconds
Elapsed time: 4507 microseconds
Elapsed time: 4240 microseconds
Elapsed time: 5034 microseconds
Elapsed time: 3998 microseconds
Elapsed time: 4009 microseconds
Elapsed time: 4030 microseconds
Elapsed time: 4071 microseconds
Elapsed time: 4096 microseconds
Elapsed time: 4053 microseconds

Requires to be connected to the target database

- For testing if you can reach the server
  - Ping / traceroute OS commands
DB2 for z/OS Timeouts

 ➔ Example: a remote client connects to DB2, does some work, and then does not go inactive (likely no COMMIT):

STC09109  DSNL027I  _DBXP SERVER DISTRIBUTED AGENT WITH
LUWID=GA65B414.PA09.1111C8065156=43494
THREAD-INFO=TOTO123:MACBOOK01:TOTO123:javaw.exe
RECEIVED ABEND=04E
FOR  REASON=00D3003B

 ➔ IDTHTTOIN
- Time, in seconds, an active server thread remains idle before it is cancelled
- Inactive connections are not subject to idle thread timeout
- In general, default works well

 ➔ Client may receive SQL30081N which would indicate that the remote connection was terminated
Client timeouts

(SQL_ATTR_QUERY_TIMEOUT)
- The client side equivalent of IDTHTOIN

Set a timeout value for SQL queries ran through the ODBC API

A timeout value of 0 disables the timeout

STC09109  DSN511I  _DBXP DSNLIENO TCP/IP CONVERSATION FAILED
TO LOCATION 10.100.1.123
IPADDR=10.100.1.124 PORT=1248
SOCKET=RECV RETURN CODE=1121 REASON CODE=00000000

QueryTimeoutInterval is the delay, in seconds, between checks for
a query timeout
- May be set larger than SQL_ATTR_QUERY_TIMEOUT
- It is a db2cli.ini keyword
How to catalog a DB2 for z/OS SSID with Client 10

→ DB2 10 for LUW does not provides the Configuration Assistant

```
DSNL080I -DZA1 DSNLTDDF DISPLAY DDF REPORT FolLOWS:
DSNL081I STATUS=STARTD
DSNL082I LOCATION LUNAME GENERICLU
DSNL083I CRISDDF USIBMT6.DDFDZA1 -NONE
DSNL084I TCPPORT=446 SECPORT=448 RESPORT=4461 IPNAME=-NONE
DSNL085I IPADDR=::10.50.1.1
DSNL086I SQL DOMAIN=CRIS59
DSNL105I CURRENT DDF OPTIONS ARE:
DSNL106I PKGREL = BndoPT
DSNL099I DSNLTDFF DISPLAY DDF REPORT COMPLETE
```

catalog tcpip node CMO9 remote 10.50.1.1 server 446 ostype mvs

catalog db CRISDDDF as DZA1 at node CMO9 authentication SERVER_ENCRYPT

catalog odbc data source DZA1
Pivot table + ODBC to connect to DB2 for z/OS

db2 => list odbc data sources

<table>
<thead>
<tr>
<th>Data source name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS Access Database</td>
<td>Microsoft Access Driver (*.mdb)</td>
</tr>
<tr>
<td>Excel Files</td>
<td>Microsoft Excel Driver (*.xls)</td>
</tr>
<tr>
<td>dBASE Files</td>
<td>Microsoft dBase Driver (*.dbf)</td>
</tr>
<tr>
<td>DZA1</td>
<td>IBM DB2 ODBC DRIVER - DB2COPY1</td>
</tr>
</tbody>
</table>
Ad-hoc reporting from a Performance Warehouse

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCEL NAME</td>
<td>DZA1STPR</td>
<td>Sum ofBYTES SENT</td>
<td>Sum ofBYTES RCD</td>
</tr>
<tr>
<td>BEGIN_REC_TSTAMP</td>
<td>Sum ofMAX QUEUED</td>
<td>Sum ofMAX QUEUED</td>
<td>Sum ofMAX QUEUED</td>
</tr>
<tr>
<td>5/13/2013 16:19</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/13/2013 16:20</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/13/2013 16:21</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/13/2013 16:22</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/13/2013 16:23</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/13/2013 16:24</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/13/2013 16:25</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5/13/2013 16:26</td>
<td>3774</td>
<td>1492</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:27</td>
<td>0</td>
<td>180</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:28</td>
<td>3774</td>
<td>1572</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:29</td>
<td>3774</td>
<td>1572</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:30</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:31</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:32</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:33</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:34</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:35</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:36</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:37</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:38</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:39</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:40</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:41</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:42</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:43</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:44</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>5/13/2013 16:45</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
</tbody>
</table>
DB2 Security options

➔ TCP/IP Already Verified (TCPALVER) subsystem parameter
  – Controls whether DB2 accepts TCP/IP connection requests that contain only a user ID
  – TCPALVER=NO → strongly recommended
  – TCPALVER=YES → RACF will not perform password checking unless the connection request sends the password
    • Potential security exposure!
  – TCPALVER=SERVER_ENCRYPT: new in DB2 10

IMPORTANT: Do not send a clear text password through the network!

➔ Consider using one of the following security options:
  – RACF PassTicket
  – Kerberos ticket
  – DRDA encrypted passwords

TIP: `db2 catalog db DB9A at node NODE1 authentication SERVER_ENCRYPT`
The resource limit tables can be used to limit the amount of resources used by dynamic queries that run on middleware servers.

Queries can be limited based on:

- Client information, including the application name, user ID, workstation ID
- IP address of the client

```sql
--DIS THD(*)
SERVER RA * 34 db2jcc_appli IDAA1 DISTSERV 008A 146248
V437-WORKSTATION=cmothink, USERID=idaa1,
APPLICATION NAME=db2jcc_application
V441-ACCOUNTING=Cris
V445-G998D43D.1775.C924C6B7E632=146248 ACCESSING DATA FOR
( 1):FFFF:9.152.212.61
V447--INDEX SESSID A ST TIME
V448--( 1) 10512:10101 W R2 1204913244630

INSERT INTO
SYSIBM.DSNRLMT01
( RLFFUNC, RLFIP, ASUTIME)
VALUES
( '8', '9.152.212.61',150000);
```

cristian@molaro.be ©® 2015
WLM client information

→ DB2 server systems have implemented the concepts of:
  – End user IDs
  – End user workstation names
  – End user application names
  – Accounting data

→ Much of this information is externalized in various forms:
  – The **DSNV437I** message of the DISPLAY THREAD command
  – THREAD-INFO data in various messages such as **DSNT375I**
  – DB2 accounting records

```
DSNL027I -PRD1 SERVER DISTRIBUTED AGENT WITH 778
LUWID=C9DE5919.F7D7.C5C2D6F15029=636
THREAD-INFO=CRIS:TotoMac:Toto:TestFromMac:*:*:*:
RECEIVED ABEND=04E
FOR REASON=00D3003B

DSNL028I -PRD1 C9DE5919.F7D7.C5C2D6F15029=636 779
ACCESSING DATA FOR
LOCATION ::10.50.1.12
IPADDR ::10.50.1.12
```
WLM client information

- Cannot use SQL for setting values
- Can use SQL for inquiring values:

```
SELECT
SUBSTR(CURRENT CLIENT_ACCTNG,1,15),
SUBSTR(CURRENT CLIENT_APPLNAME,1,15),
SUBSTR(CURRENT CLIENT_USERID,1,15),
SUBSTR(CURRENT CLIENT_WRKSTNNAME,1,15)
FROM SYSIBM.SYSDUMMY1;
```

- The value of these special register can be changed by using:
  - `SQLE_CLIENT_INFO_USERID` (sqleseti)
  - `DB2Connection.setDB2ClientUser(String info)`
  - The RRS DSNRLI SIGNON, AUTH SIGNON, CONTEXT SIGNON, or SET_CLIENT_ID function
  - The `WLM_SET_CLIENT_INFO` stored procedure
WLM_SET_CLIENT_INFO

SYNTAX:
```bash
>>-WLM_SET_CLIENT_INFO-(-+-client_userid-+-,-+-client_wrkstnname-+-,->
                        'NULL------------'-NULL------------'
                        >+-client_applname-+-,-+-client_acctstr-+-)------------>
                        'NULL------------'-NULL------------'
```

→ Korn shell script example

```bash
#!/usr/bin/ksh
....
client_uid="cristian"
client_wrkst="Office Desktop"
client_appl="RC01"
client_acc="RC01"
....

db2 +o "Connect to " $MFDB2 " user " $HOSTuser " using " $HOSTpasswd

db2 "CALL SYSPROC.WLM_SET_CLIENT_INFO('"$client_uid"','"$client_wrkst"
                              ,"$client_appl"','"$client_acc"');"

db2 -xtof /home/cognos/scripts/queries/RC01
....

# End program
```
WLM_SET_CLIENT_INFO cost

```
TIMES/EVENTS        APPL(CL.1)      DB2(CL.2)
--------------------        ------------        ------------
ELAPSED TIME           0.015367        0.004823
NONNESTED             0.007966         0.000505
STORED PROC           0.007402         0.004319
UDF                   0.000000         0.000000
TRIGGER               0.000000         0.000000
CP CPU TIME           0.001771         0.001340
AGENT                 0.001771         0.001340
NONNESTED             0.000509         0.000397
STORED PRC            0.001262         0.000943
UDF                   0.000000         0.000000
TRIGGER               0.000000         0.000000
PAR.TASKS             0.000000         0.000000

Up to 20 mSecs in first exec
```

db2 "CALL SYSPROC.WLM_SET_CLIENT_INFO(  
    '"$clientuid'",  
    '"$clientwrktn'",  
    '"$clientapp'",  
    '"$report'"');"

db2 -xto $stmt1
db2 -xtof /home/cognos/scripts/RI09
db2 +o terminate

dsnadmsi

```
DSNADMSI                VALUE                DSNADMSI                TIMES                DSNADMSI                TIME
--------------------                -------------                -----------                -------------                -------
TYPE                   PACKAGE           ELAPSED TIME - CL7      0.000012            LOCK/LATCH               0.000000
LOCATION               DWHDA12           CP CPU TIME              0.000010            IRLM LOCK+LATCH           0.000000
COLLECTION ID           DSNADMSI         AGENT                     0.000010            DB2 LATCH                 0.000000
PROGRAM NAME           DSNADMSI         PAR.TASKS                 0.000000            SYNCHRONOUS I/O           0.000000
CONSISTENCY TOKEN       0E5F1F0D09F14040 SE CPU TIME               0.000000            OTHER READ I/O            0.000000
ACTIVITY TYPE           STORED PROC      SUSPENSION-CL8         0.000000            OTHER WRITE I/O           0.000000
ACTIVITY NAME           WLM_SET_CLIENT_I#1 AGENT                     0.000000            SERV.TASK SWITCH          0.000000
SCHEMA NAME             SYSPROC           PAR.TASKS                 0.000000            ARCH.LOG(QUIESCE)         0.000000
SUCC AUTH CHECK         NO                NOT ACCOUNTED            0.000001            ARCHIVE LOG READ          0.000000
NBR OF ALLOCATIONS      1                NOT ACCOUNTED            0.000000            DRAIN LOCK                0.000000
SQL STMT - AVERAGE      1.00              CP CPU SU                 1                CLAIM RELEASE            0.000000
SQL STMT - TOTAL        1                AGENT                     1                PAGE LATCH                0.000000
NBR RLUP THREADS        1                PAR.TASKS                 0                NOTIFY MESSAGES           0.000000
```
Accounting information and WLM

- Client info can be used to classify work in WLM
- WLM Classification rules
  - For work type DDF: DDF Accounting information is the value of the DB2 accounting string associated with the DDF server thread

* Subsystem Type DDF – Distributed Workload

Classification:

Default service class is **SCDDFLOW**
There is no default report class.

<table>
<thead>
<tr>
<th>Qualifier name</th>
<th>Starting position</th>
<th>Service Class</th>
<th>Report Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>DA12*</td>
<td></td>
<td>SCREPLO</td>
<td>RCUNKWN</td>
</tr>
<tr>
<td>. RC01*</td>
<td>56</td>
<td>SCREPLO</td>
<td>RCRC01</td>
</tr>
<tr>
<td>. RS02*</td>
<td>56</td>
<td>SCREPHI</td>
<td>RCRS02</td>
</tr>
<tr>
<td>. RC03*</td>
<td>56</td>
<td>SCREPLO</td>
<td>RCRC03</td>
</tr>
<tr>
<td>. RS04*</td>
<td>56</td>
<td>SCREPHI</td>
<td>RCRS04</td>
</tr>
<tr>
<td>. RS05*</td>
<td>56</td>
<td>SCREPHI</td>
<td>RCRS05</td>
</tr>
<tr>
<td>. RS06*</td>
<td>56</td>
<td>SCREPHI</td>
<td>RCRS06</td>
</tr>
<tr>
<td>. RI09*</td>
<td>56</td>
<td>SCREPMD</td>
<td>RCRI09</td>
</tr>
<tr>
<td>. RI10*</td>
<td>56</td>
<td>SCREPMD</td>
<td>RCRI10</td>
</tr>
</tbody>
</table>
Can you tell the difference?
Protecting the work that matters

➤ WLM can help to
  ➤ Protect critical workloads
  ➤ Provide consistent response times
➤ Example:
DDF and Classification Rules

- Classification rules used to assign the incoming work to a
  - Service Class
  - Reporting Class (optional)
- If you do not implement classification rules for DDF
  - All DDF workload is classified and executed under the service class…

**SYSOTHER**

- System provided service class for all work not associated with a service class
- It is assigned a *discretionary goal*
- *Discretionary work is run using any system resources not required to meet the goals of other work*
ACCOUNTING AND SMF
DB2 Accounting rollup

- DB2 writes an accounting record when a DDF thread:
  - Ends
  - Is made inactive
  - Does not go inactive because using KEEPDYNAMIC(YES)
  - Or a sign-on occurs for an RRSAF thread
- This has the potential to create a huge amount of records
- zParm ACCUMACC controls whether and when DB2 accounting data is accumulated for DDF and RRSAF threads
  - ACCUMACC=NO, default no effect
  - ACCUMACC = n, (n defines the accumulation interval)
- zParm ACCUMUID defines the aggregation criteria
  - Value from 0 to 17
  - ACCUMUID=1 → End user ID
Accounting interval and distributed threads

- Accounting record cut when the connection goes inactive
  - CMTSTAT = INACTIVE
Accounting rollup

- Accounting rollup accumulates accounting information
  - DDF and RRS only
DB2 10 introduces SMF compression

- Controlled by new system parameter SMFCOMP
  - OFF (default): SMF trace records are not compressed
  - ON: Trace records written to SMF are compressed

- The z/OS compression service CSRCESRV compresses everything after the SMF header

- Data Sharing environment: SMFCOMP has member scope

- Performance measurements
  - Minimal overhead; ~ 1% with Accounting Class 1, 2, 3, 7, 8, 10
  - The disk savings for DB2 SMF data set can be significant with compression rate of 60% - 80%

- APAR PM27872
  - Decompression routine DSNTSMFD
  - Sample JCL DSNTEJDS

**TIP:** SMF Compression is preferable to Accounting Rollup
HIGH PERFORMANCE DBATS
Benefits of HP DBAT + RELEASE(DEALLOCATE)

- DB2 10 High Performance DBAT support reduces CPU consumption by:
  - Supporting RELEASE(DEALLOCATE)
  - Avoid repeated package allocation/de-allocation
  - Avoids acquiring and releasing parent (IS, IX) locks frequently
  - Avoids the processing necessary to go INACTIVE and then back to ACTIVE
  - More noticeable CPU reduction for short transactions

- Behavior
  - DBAT will stay associated with connection at UOW boundaries if there is at least one RELEASE(DEALLOCATE) package allocated
  - DBAT will be terminated after 200 uses
  - Normal idle thread time-out IDTHTOIN detection will be applied to these DBATs

**TIP:** No benefit and not support for ACTIVE threads (CMSTATS=ACTIVE)

**TIP:** No benefit for KEEPDYNAMIC YES
RELEASE(COMMIT) vs RELEASE(DEALLOCATE)

<table>
<thead>
<tr>
<th>Total CPU per transaction</th>
<th>V9</th>
<th>V10 PKREL(COMMIT)</th>
<th>Delta (%)</th>
<th>V10 PKREL(BNDOPT)</th>
<th>Delta (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQCL</td>
<td>2114</td>
<td>1997</td>
<td>-5.5</td>
<td>1918</td>
<td>-9.3</td>
</tr>
<tr>
<td>SPCB</td>
<td>1221</td>
<td>1124</td>
<td>-7.9</td>
<td>1056</td>
<td>-13.5</td>
</tr>
<tr>
<td>JDBC</td>
<td>2152</td>
<td>2017</td>
<td>-6.3</td>
<td>1855</td>
<td>-13.8</td>
</tr>
<tr>
<td>SQLJ</td>
<td>1999</td>
<td>1761</td>
<td>-11.9</td>
<td>1689</td>
<td>-15.5</td>
</tr>
<tr>
<td>SPSJ</td>
<td>1759</td>
<td>1642</td>
<td>-6.7</td>
<td>1550</td>
<td>-11.9</td>
</tr>
<tr>
<td>SPNS</td>
<td>1472</td>
<td>1304</td>
<td>-11.4</td>
<td>1180</td>
<td>-19.8</td>
</tr>
</tbody>
</table>

- Total CPU per txn = System Services Address Space + Database Services Address Space + IRLM + DDF Address Space CPU
- CPU time in microseconds.
Exploiting High Performance DBATs

→ To enable:

- REBIND with RELEASE(DEALLOCATE)
- MODIFY DDF PKGREL (BNDOPT)

→ To disable:

- MODIFY DDF PKGREL (COMMIT) to overlaid BNDOPT option

→ To monitor:

- Statistics GLOBAL DDF activity report

<table>
<thead>
<tr>
<th>GLOBAL DDF ACTIVITY</th>
<th>QUANTITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUR ACTIVE DBATS-BND DEALLC</td>
<td>5.39</td>
</tr>
<tr>
<td>HWM ACTIVE DBATS-BND DEALLC</td>
<td>10.00</td>
</tr>
</tbody>
</table>
### DIS DDF DETAIL in DB2 10

<table>
<thead>
<tr>
<th>LOCATION</th>
<th>LUNAME</th>
<th>GENERICLU</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB0A</td>
<td>USIBMSC.SCPDB0A</td>
<td>-NONE</td>
</tr>
</tbody>
</table>

TCP PORT = 12345, SECPORT = 12346, RESPORT = 12347, IPNAME = -NONE

IP ADDR = ::10.50.1.1

SQL DOMAIN = wtsc63.itso.ibm.com

<table>
<thead>
<tr>
<th>ALIAS</th>
<th>PORT</th>
<th>SECPORT</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>0</td>
<td>0</td>
<td>STOPD</td>
</tr>
<tr>
<td>TEST</td>
<td>0</td>
<td>0</td>
<td>STOPD</td>
</tr>
<tr>
<td>TEST2</td>
<td>0</td>
<td>0</td>
<td>STOPD</td>
</tr>
</tbody>
</table>

DT=I CONDBAT = 10000, MDBAT = 200
ADBAT = 0, QUEDBAT = 0, INADBAT = 0, CONQUED = 0
DSCDBAT = 0, INACONN = 1

**CURRENT DDF OPTIONS ARE:**

- PKGREL = COMMIT

---

**Diagram:**

- MODIFY DDF
- ALIAS(alias-name) ADD
- DELETE
- START
- BNDOPT
- PKGREL(COMMIT)
Implement gradually or selectively

- BIND a new set of packages with RELEASE(DEALLOCATE)
- SYSSHxyy
  - S: Represents a small package (65 sections)
  - H: Represents WITH HOLD
  - x: Indicates the isolation level
    - 1=UR, 2=CS, 3=RS, 4=RR
  - yy: The package iteration 00 through FF
Release Deallocate Enhancements in DB2 11

- RELEASE(DEALLOCATE) Bind/Rebind option
  - Avoid package allocation overhead
  - CPU savings with transactions with frequent commits

- Concerns
  - REBIND, DDL and online REORG cannot break-in with local persistent threads using RELEASE(DEALLOCATE)

- DB2 11
  - Allows REBIND/DDL, and online REORG to break in “committed” persistent threads with REL(DEALLOCATE)

TIP: More aggressive adaptation of RELEASE(DEALLOCATE) is possible

PM95929: BREAK INTO PERSISTENT THREAD DOES NOT BREAK LOCALLY ATTACHED THREAD WITH RELEASE(DEALLOCATE) FOR BIND AND DDL
Break-in into High Performance DBATs

To enable HP DBATs in DB2 10 and DB2 11:

- Create a collection of packages with RELEASE(DEALLOCATE)
  - Do NOT bind NULLID col. with RELEASE(DEALLOCATE)
- Modify client applications to request packages from a different collection via CURRENTPACKAGESET
- Issue \texttt{MODIFY DDF PKGREL(BNDOPT)}

To disable:

- Issue \texttt{MODIFY DDF PKGREL(COMMIT)}
- Existing running DBATs will be terminated on next COMMIT
- Idle DBATs waiting for a new transaction will be terminated during the next two minutes DDF
- New DBATs will only allocate packages in RELEASE(COMMIT)

DB2 11 break-in

- Automatically done on next COMMIT if waiter on a package lock
CLOSING
There is *more* that you *must* know!

- Automatic client reroute and work load balancing
- Stored procedures
- Autocommit implications
- Tracing distributed applications
- Application development best practices
- Security topics
- Trusted contexts and roles
- …

In the appendix
  - Data Sharing considerations
  - Specialty engine considerations
  - What is new in DB2 11?
  - Problem determination
This was our Agenda

**Objective**
- To review topics that you *must* know about distributed access to DB2 for z/OS

**Contents**
- Connectivity topics
- Accounting and SMF
- High performance DBATs
- And there is more
- **In Part 2**
  - Specialty engines
  - What is new in DB2 11
  - Problem determination
Questions?

CRISTIAN@MOLARO.BE
THANKS!

CRISTIAN MOLARO

cristian@molaro.be
@cristianmolaro