PDUG

Db2 zOS 13 Application Development Topics

Michal Bialecki – BROADCOM

credits to Emil Kotrc

POLISH DB2 USERS GROUP

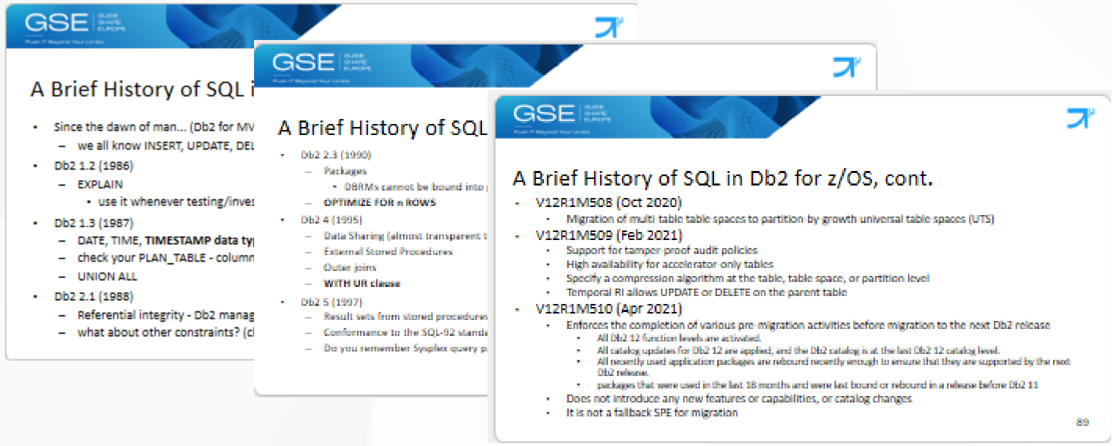
AGENDA

- Continuous Delivery
- SQL Changes in Db2 13
- Application level lock control (Multi-tenancy)
- Profile Table Enhancements for local applications

- Appendix
  - A Brief History (SQL Changes up to Db2 12)

DB2 13 APPLICATION DEVELOPMENT TOPICS?

- We can start with DB2 1 ☺
- However, will only review what's new in Db2 13
- For more on SQL features history in Db2 see the Appendix



A Brief History of SQL i

- Since the dawn of man... (Db2 for MV
  - we all know INSERT, UPDATE, DEL
- Db2 1.2 (1986)
  - EXPLAIN
    - use it whenever testing/inves
- Db2 1.3 (1987)
  - DATE, TIME, **TIMESTAMP data ty**
  - check your PLAN_TABLE - column
  - UNION ALL
- Db2 2.1 (1988)
  - Referential integrity - Db2 manag
  - what about other constraints? (cl

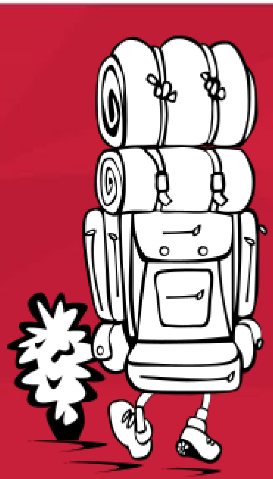
A Brief History of SQL

- Db2 2.3 (1990)
  - Packages
    - DBRMs cannot be bound into
  - **OPTIMIZE FOR n ROWS**
- Db2 4 (1995)
  - Data Sharing (almost transparent t
  - External Stored Procedures
  - Outer joins
  - **WITH UR clause**
- Db2 5 (1997)
  - Result sets from stored procedures
  - Conformance to the SQL-92 standa
  - Do you remember Sysplex query p

A Brief History of SQL in Db2 for z/OS, cont.

- V12R1M508 (Oct 2020)
  - Migration of multi-table table spaces to partition-by-growth universal table spaces (UTS)
- V12R1M509 (Feb 2021)
  - Support for tamper-proof audit policies
  - High availability for accelerator-only tables
  - Specify a compression algorithm at the table, table space, or partition level
  - Temporal RI allows UPDATE or DELETE on the parent table
- V12R1M510 (Apr 2021)
  - Enforces the completion of various pre-migration activities before migration to the next Db2 release
    - All Db2 12 function levels are activated.
    - All catalog updates for Db2 12 are applied, and the Db2 catalog is at the last Db2 12 catalog level.
    - All recently used application packages are rebound recently enough to ensure that they are supported by the next Db2 release.
    - packages that were used in the last 18 months and were last bound or rebound in a release before Db2 11
  - Does not introduce any new features or capabilities, or catalog changes
  - It is not a fallback SPE for migration

89



CONTINUOUS DELIVERY

CONTINUOUS DELIVERY

- Catalog Level, Function Level, Maintenance Level (code level)
  -DIS GROUP:

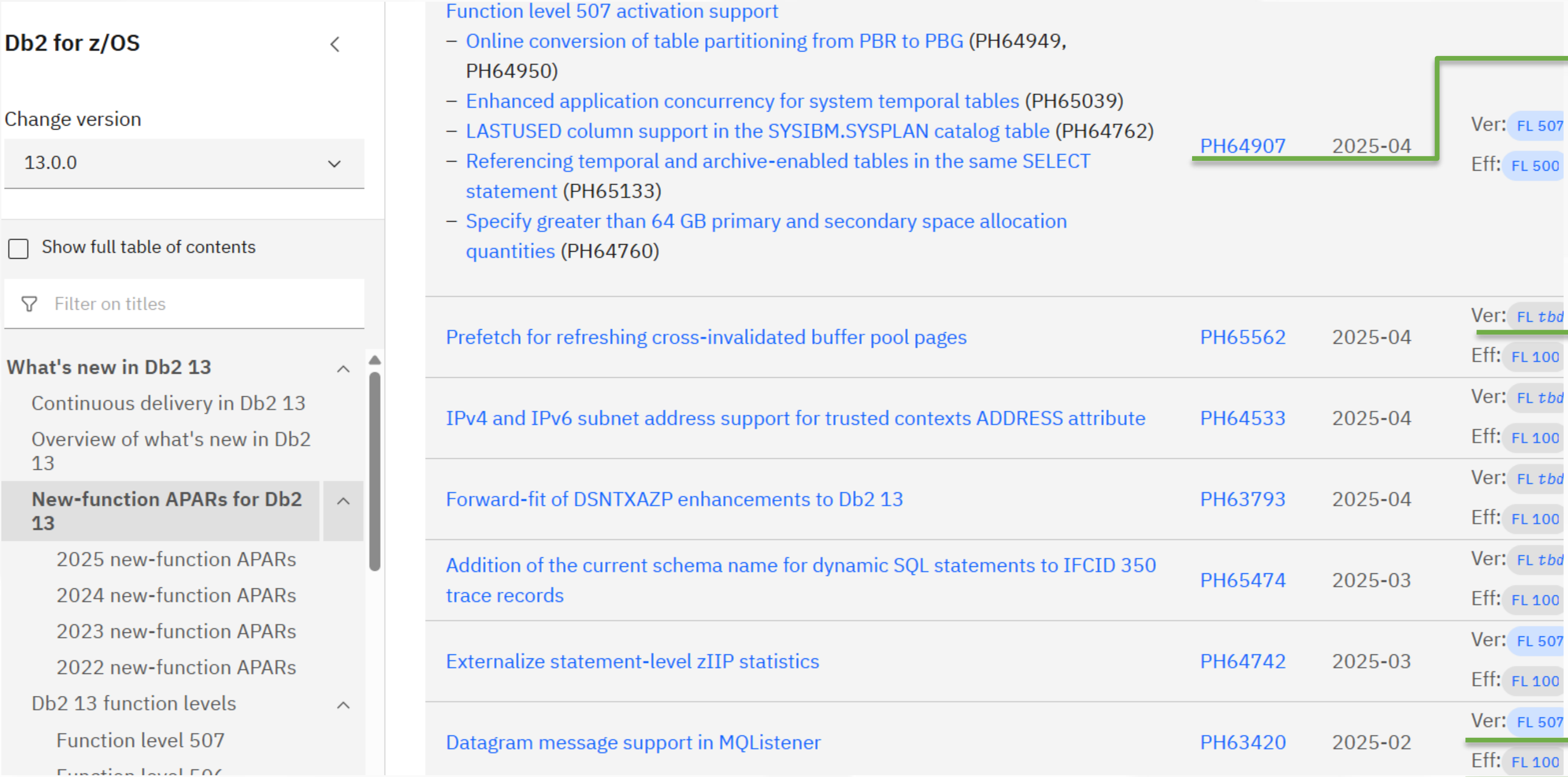
```
*** BEGIN DISPLAY OF GROUP(........) CATALOG LEVEL(V13R1M501)
                  CURRENT FUNCTION LEVEL(V13R1M501)
                  HIGHEST ACTIVATED FUNCTION LEVEL(V13R1M501)
                  HIGHEST POSSIBLE FUNCTION LEVEL(V13R1M502)
                  PROTOCOL LEVEL(2)
                  GROUP ATTACH NAME(....)
---------------------------------------------------------------------
DB2          SUB                     DB2    SYSTEM    IRLM
MEMBER   ID  SYS  CMDPREF   STATUS   LVL    NAME      SUBSYS IRLMPROC
-------- --- ---- --------  -------- ------ --------  ----   --------
........   0 ssid !ssid     ACTIVE   131502 lpar      irlm   ssidIRLM
---------------------------------------------------------------------
```

- APPLCOMPAT (bind)
- SET CURRENT APPLICATION COMPATIBILITY
  - In application
  - or by system profile
- clientApplcompat - driver client property

CONTINUOUS DELIVERY

- The same concept as we have learned in Db2 12, differs a bit in detail:
  - in V12 - many functions delivered outside of Function Levels (FL)
    - implications:
      - Various combinations of functionality was possible independent of function levels activated
  - in V13 - all non-tied to Function Level APARs are PRErequisites to “some” Function Level enabling APAR:
    - implications:
      - activation of specific Function Level guarantees that all new functions are available, up to FL cut-off
        - It is called *“verifying function level”*
      - helps with identifying what function is available and what is not, regardless if APAR was tied to function level, without detailed PTF checks (eg from MEPL)

» https://www.ibm.com/docs/en/db2-for-zos/13.0.0?topic=13-new-function-apars-db2



Db2 for z/OS

Change version

13.0.0

□ Show full table of contents

Filter on titles

What's new in Db2 13
- Continuous delivery in Db2 13
- Overview of what's new in Db2 13
- **New-function APARs for Db2 13**
  - 2025 new-function APARs
  - 2024 new-function APARs
  - 2023 new-function APARs
  - 2022 new-function APARs
- Db2 13 function levels
  - Function level 507
  - Function level 506

Function level 507 activation support – Online conversion of table partitioning from PBR to PBG (PH64949, PH64950) – Enhanced application concurrency for system temporal tables (PH65039) – LASTUSED column support in the SYSIBM.SYSPLAN catalog table (PH64762) – Referencing temporal and archive-enabled tables in the same SELECT statement (PH65133) – Specify greater than 64 GB primary and secondary space allocation quantities (PH64760)	PH64907	2025-04	Ver: FL 507 Eff: FL 500
Prefetch for refreshing cross-invalidated buffer pool pages	PH65562	2025-04	Ver: FL *tbd* Eff: FL 100
IPv4 and IPv6 subnet address support for trusted contexts ADDRESS attribute	PH64533	2025-04	Ver: FL *tbd* Eff: FL 100
Forward-fit of DSNTXAZP enhancements to Db2 13	PH63793	2025-04	Ver: FL *tbd* Eff: FL 100
Addition of the current schema name for dynamic SQL statements to IFCID 350 trace records	PH65474	2025-03	Ver: FL *tbd* Eff: FL 100
Externalize statement-level zIIP statistics	PH64742	2025-03	Ver: FL 507 Eff: FL 100
Datagram message support in MQListener	PH63420	2025-02	Ver: FL 507 Eff: FL 100

FL 507 enabling APAR

Verifying Function Level not set yet - APAR was released after FL507

Verifying Function Level is set to FL507 and its’ activation “guarantees” the PTF is applied and its’ new function is available

Minimum (Effective) Function level when APAR can be applied and takes effect, independent from Veryfying Function Level

DB2 13 FUNCTION LEVELS - SUMMARY

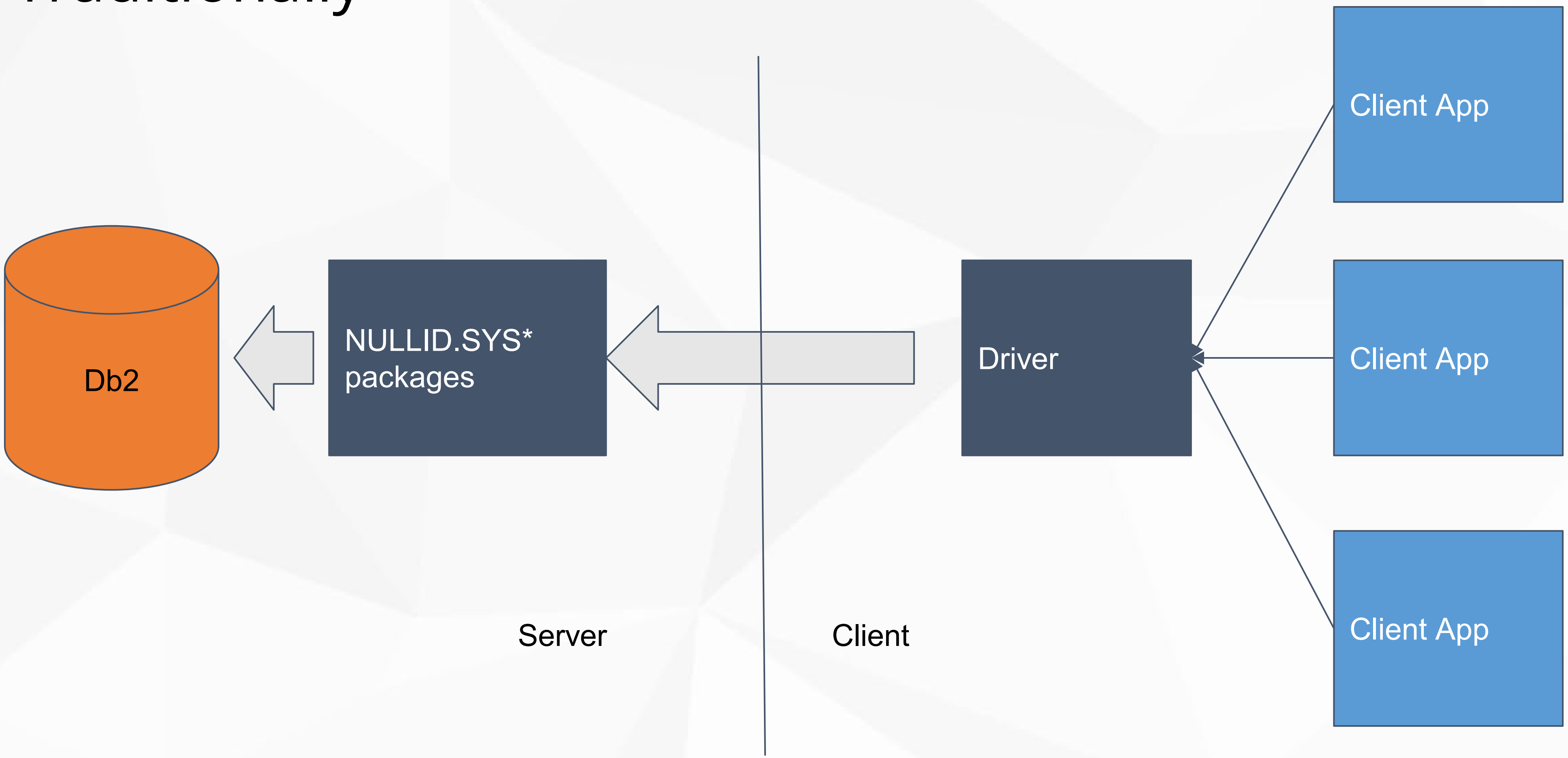
Function Level	Catalog Level change	Incompatible changes	SQL Changes	Notes
V13R1M100	Yes (= V12R1M509)	Yes	Yes	
V13R1M500	No	No	Yes	New functions that do not require catalog changes
V13R1M501	Yes	No	Yes	New functions that require catalog changes Newly installed system starts here or below
V13R1M502	No	No	No	
V13R1M503	No	No	Yes	
V13R1M504	Yes	Yes	Yes	
V13R1M505	Yes	No	Yes	
V13R1M506	No	No	Yes	
V13R1M507	Yes	No	Yes	New functions that require catalog changes

APPLCOMPAT AND DISTRIBUTED PACKAGES

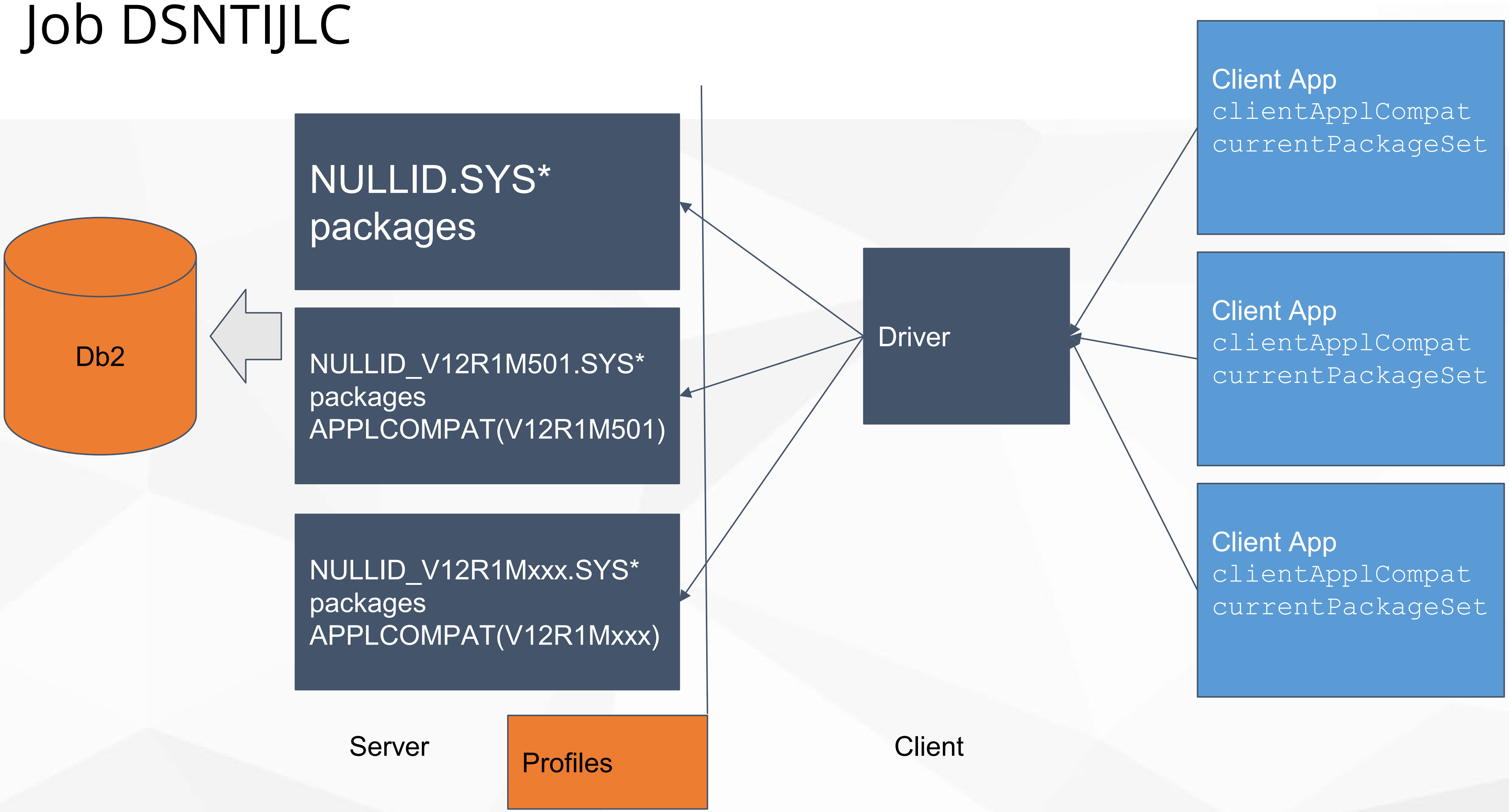
- Compatible drivers: Data Server Driver 11.1 Modification 2 Fix Pack 2 if you go beyond V12R1M500
  - **no need to set the clientApplcompat** property regardless of APPLCOMPAT
    - prior to V12 PH08482, Db2 was rejecting the client connection with SQLCODE -30025
- General recommendation – two sets of Data Server Driver packages (NULLID):
  - Stabilized at a certain minimal level (ex. NULLID.SYS*)
  - New SQL (ex. NULLID_V13R1M501.SYS*)

1. Create new driver packages – Db2Binder or **DSNTIJLC**
2. Enable to use the right collection
   - Client property (**currentPackageSet**)
   - Or **Profile tables**

» Traditionally



- Recommendation after activating V12R1M501
- Job DSNTIJLC



DB2 13 SQL CHANGES

Allow SQL to reference a temporal table and archive-enabled table in the same statement– V13R1M507

```
SELECT ATT.POLICY_ID
FROM
     application_period_temporal_table ATT,
     system-period temporal table STT,
     archive_enabled_table AET
WHERE      ATT.CLIENT_ID = STT.CLIENT_ID
     AND ATT.CLIENT_ID = AET.CLIENT_ID;
```

The above statement will succeed if CURRENT TEMPORAL BUSINESS_TIME is NULL, CURRENT TEMPORAL SYSTEM_TIMEis NULL, and SYSIBMADM.GET_ARCHIVE is set to ‘N’ = no historical data is intended to be selected

Prior, when an SQL statement references both a temporal table and archive-enabled table, Db2 z/OS issued SQLCODE -20555 with reason-code 1

INSERT WITH MULTI-ROW VALUES CLAUSE – V13R1M507

```
INSERT INTO DSN8D10.EMP
(EMPNO,FIRSTNME,MIDINIT,LASTNAME,WORKDEPT,PHONENO,
                         HIREDATE,JOB,EDLEVEL,SEX,BIRTHDATE,
                         SALARY,BONUS,COMM)
   VALUES ('000206','ELIZABETH','T','GRACE','D11','2866',
            '2023-02-17','ANALYST',16,'F','1975-03-21',
           16345,500,2300),
           ('000207','JACK','Q','JOHNSON','D11','2867',
            '2023-08-10','ANALYST',16,'M','1979-07-22',
           16345,500,2300),
           ('000208','JENNIFER','K','WHITE','D11','2868',
            '2023-08-10','ANALYST',16,'F','1980-08-13',
           16345,500,2300);
```

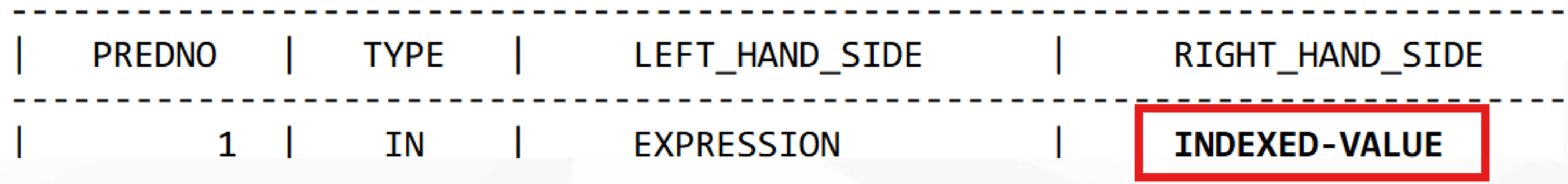
ADD IMPLICITLY HIDDEN ROWID COLUMNS – V13R1M506

- Useful for converting existing tables to a partition-by-range (PBR) when a suitable column does not already exist for the partitioning key
  - Allows balanced spread of values across partitions
  - With no natural clustering
- `ALTER TABLE ttt ADD COLUMN ccc` **`IMPLICITLY HIDDEN ROWID`** `GENERATED BY DEFAULT`
  - Fails with SQLCODE -270 below V13R1M506
  - This restriction is removed with V13R1M506 or higher.
- Column is transparent to application (as implicitly hidden)

IMPROVED PERFORMANCE FOR IN-LIST PREDICATES – V13R1M506

```
SELECT * FROM T1 WHERE INT(C2) IN (3.14, 2+2, 44, 105, 47, 21, 3, 1975);
```

Right hand side (RHS) of predicate starts with a non-integer literal value (i.e. 3.14), then followed by all integer values, which now Db2 can consistent datatypes (eg DECIMAL(4,2))
predicates are not stage1 (sargable) or indexable, but now allows Db2 to compare left hand side (LHS) with all items on RHS through new hashing algorithm, which makes the performance comparable to a matching predicate in an index search.



```
------------------------------------------------------------------------------
|   PREDNO    |   TYPE   |       LEFT_HAND_SIDE       |      RIGHT_HAND_SIDE
------------------------------------------------------------------------------
|           1 |    IN    |       EXPRESSION           |      INDEXED-VALUE
```

INTERPRET SCALAR FUNCTION – V13R1M505

- Interprets binary string as the specified data type
- Example:
  - **DSNU733I csect-name ROW (RID=X'00070070EC8104') HAS NO PARENT FOR relationship-identifier**
  - **INTERPRET(BX'0000070070EC8104' AS BIGINT) = 7698475942148**
  - can be used in select … WHERE RID(*table1*) = 7698475942148

- Example 2: to check how data looks in other encoding
  - **INTERPRET(BX'616263' AS CHAR(3) CCSID 37)   /ÃÄ**
  - **INTERPRET(BX'616263' AS CHAR(3) CCSID 1208)   abc**

STATEMENT-LEVEL INVALIDATION – V13R1M504

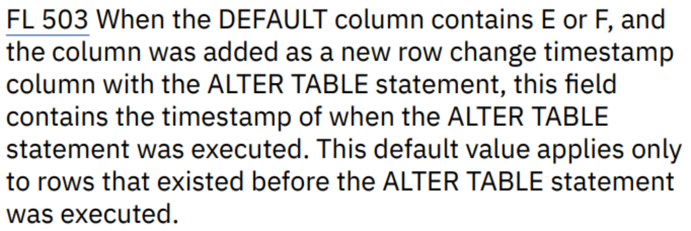
- In V13R1M502, Db2 introduced new **DEPLEVEL**(STATEMENT) bind option and new ZPARM **PACKAGE_DEPENDENCY_LEVEL**, which was in preparation of statement-level invalidation enabled in **V13R1M504**
- Applications can **execute** a package that is **invalidated** at a statement level without waiting for the completion of the autobind of a statement
- **Valid** statements can be executed **immediately**
- **Invalid** statements go through incremental bind
  - If incremental bind fails, it may result in different SQLCODEs – incompatible change
- **Autobind phase-in –** new threads use the new valid copy of the package
  - Similar to rebind phase-in but does not require PLANMGMT EXTENDED. ABIND must be YES/COEXIST
    - The automatic rebind value for APREUSE is WARN, and APREUSESOURCE is CURRENT

      = WARN me if my access path changes, try to preserve access path using current .
- **WARNING:** unneeded dependency records for invalidated statements in the SYSPACKDEP and SYSPACKSTMTDEP tables remains, needs to be removed with FREE PACKAGE command with the PLANMGMTSCOPE(PHASEOUT)

ALTER TABLE WITH ADD COLUMN FOR ROW CHANGE TIMESTAMP - V13R1M503

- Default value for **existing rows** is now set to the **timestamp of the ALTER**
- Row change timestamp introduced in Db2 9
  - Default value was derived from the row's **page header** (RBA or LRSN),
  - on page we have many rows.. this value for non-changed rows was **changing with updates** to other rows to that page
- Behavior should now be predictable in V13R1M503
- Default saved in SYSIBM.SYSCOLUMNS.**DEFAULTVALUE**
- Very useful for detection of changes:

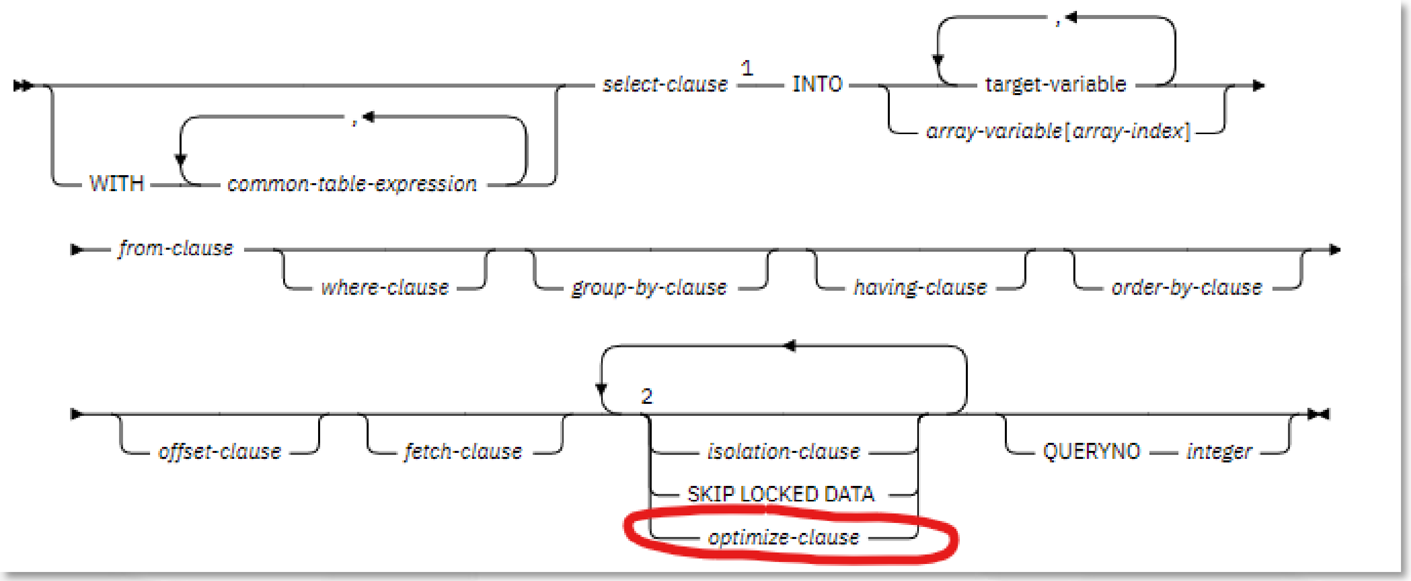
```
SELECT COUNT(*) FROM EMP WHERE
ROW CHANGE TIMESTAMP FOR EMP > CURRENT TIMESTAMP - 1 YEAR;
```



FL 503 When the DEFAULT column contains E or F, and the column was added as a new row change timestamp column with the ALTER TABLE statement, this field contains the timestamp of when the ALTER TABLE statement was executed. This default value applies only to rows that existed before the ALTER TABLE statement was executed.

OPTIMIZE FOR SELECT INTO - V13R1M503

- OPTIMIZE FOR n ROWS can be specified for SELECT INTO
  - To **influence the access path**
  - This is singleton anyway, right?
- OPTIMIZE FOR 1 ROW - **sort avoidance** preference
- OPTIMIZE FOR 2 ROWS - consider access path that **uses a sort**



DEADLOCK_RESOLUTION_PRIORITY - V13R1M501

- Allow applications to specify a **deadlock resolution priority**
  - **SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY** built-in global variable
- **Profile table** enhancements for application environment settings
  - Specify values for **SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY**
- These topics will be covered in extra sections

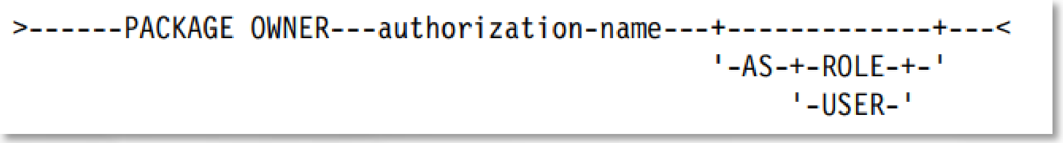
CURRENT LOCK TIMEOUT - V13R1M500

- Increased control for applications over how long to wait for a lock
  - **CURRENT LOCK TIMEOUT** special register and the **SET CURRENT LOCK TIMEOUT** SQL statement
- Profile table enhancements for application environment settings
  - specify values to **CURRENT LOCK TIMEOUT** special register
- SQL Data Insights
- Will be covered in an extra sections

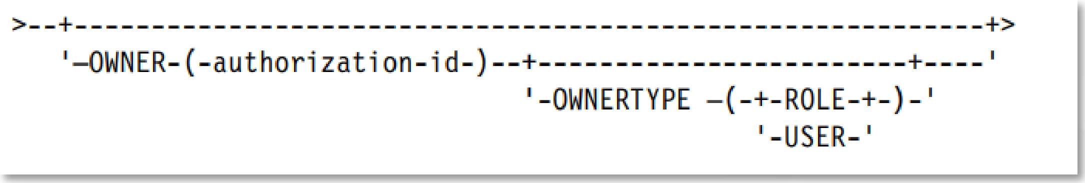
FLEXIBILITY FOR PACKAGE OWNERSHIP – V13R1M500

- New clauses **AS ROLE** or **AS USER** for
  - ALTER/CREATE FUNCTION - SQL scalar
  - ALTER/CREATE PROCEDURE – Native SQL
- **OWNERTYPE** BIND/REBIND option
- Example:
  - DBA under trusted context binding a package
  - Role as an object owner
  - Package to be owned by an authorization ID

  - Later, change this to be role:



```
>------PACKAGE OWNER---authorization-name---+-------------+---<
                                            '-AS-+-ROLE-+-'
                                                 '-USER-'
```



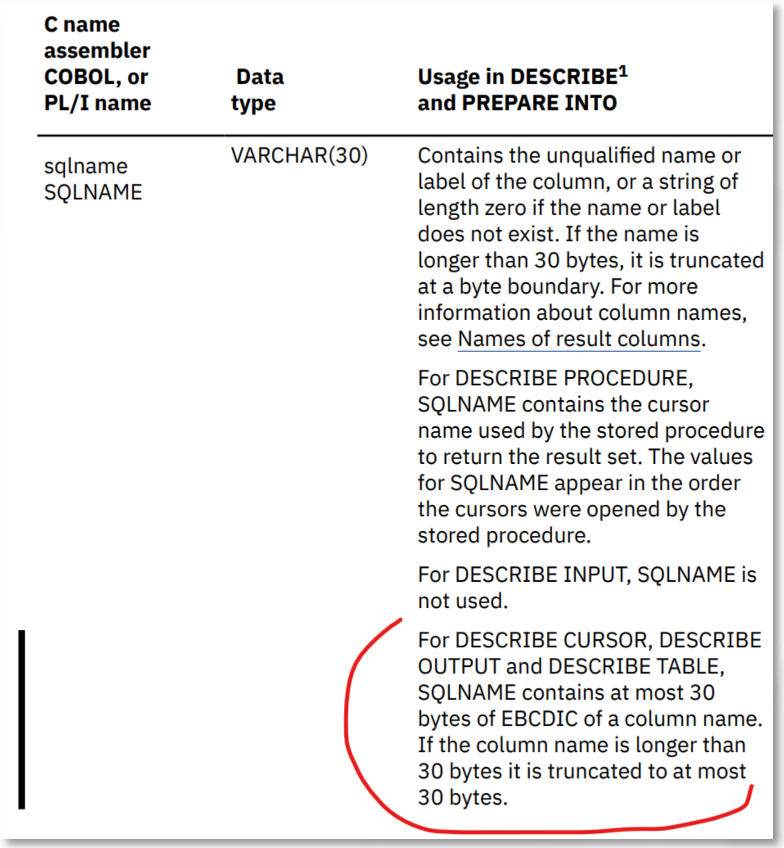
```
>--+----------------------------------------------------------+>
   '–OWNER-(-authorization-id-)--+------------------------+----'
                                 '-OWNERTYPE –(-+-ROLE-+-)-'
                                                '-USER-'
```

```
BIND PACKAGE(MYCOLLID) MEMBER(MYPKG)OWNER(USER01) OWNERTYPE(USER)
```

```
REBIND PACKAGE(MYCOLLID.MYPKG) OWNER(DEVROLE) OWNERTYPE(ROLE)
```

COLUMN NAMES LONGER THAN 30 CHARACTERS – V13R1M100

- I would be super careful with this
  - Application impact
  - Vendor tools impact
- From **30 bytes** EBCDIC up to **128** characters
- **TABLE_COL_NAME_EXPANSION**
  - OFF is default
- Column names in **SQLDA** are 30 bytes –
  - WARNING: truncation
  - Targets mainly distributed applications



C name assembler COBOL, or PL/I name	Data type	Usage in DESCRIBE[1] and PREPARE INTO
sqlname SQLNAME	VARCHAR(30)	Contains the unqualified name or label of the column, or a string of length zero if the name or label does not exist. If the name is longer than 30 bytes, it is truncated at a byte boundary. For more information about column names, see Names of result columns. For DESCRIBE PROCEDURE, SQLNAME contains the cursor name used by the stored procedure to return the result set. The values for SQLNAME appear in the order the cursors were opened by the stored procedure. For DESCRIBE INPUT, SQLNAME is not used. For DESCRIBE CURSOR, DESCRIBE OUTPUT and DESCRIBE TABLE, SQLNAME contains at most 30 bytes of EBCDIC of a column name. If the column name is longer than 30 bytes it is truncated to at most 30 bytes.

APPLICATION LEVEL LOCK CONTROL

PROBLEM DEFINITION – BASIC DEFINITIONS

- Db2 for z/OS **concurrency control** is lock based
  - Compare it to multi-versioning (MVCC, non-locking concurrency control)
- This is to provide full **ACID** properties
  - **A**tomicity
  - **C**onsistency
  - **I**solation
  - **D**urability
- Db2 uses **locks** to provide transaction isolation, concurrency, and serialization

LOCK CHARACTERISTICS

- Lock **size**
  - Table space, Table, Partition, Page, Row, LOB, XML
- **Duration**
  - length of time the lock is held
- **Mode**
  - Page and row – Share, Update, eXclusive
  - Partition, table space, table – IS (IntentShare), IX (IntentEXclusive), S (share), U(Update), SIX (ShareIntentEXclusive), X(eXclusive)

LOCKS COMPATIBILITY

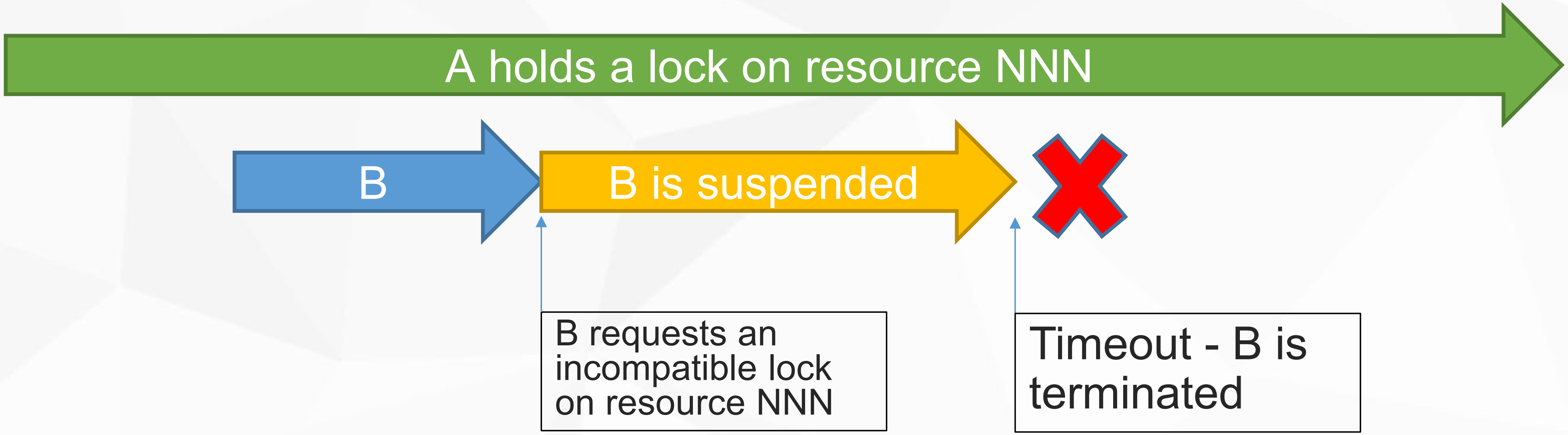
PAGE and ROW locks	S-lock	U-lock	X-lock
S-lock	Yes	Yes	No
U-lock	Yes	No	No
X-lock	No	No	No

Table, table space (or partition)	IS	IX	S	U	SIX	X
IS	Yes	Yes	Yes	Yes	Yes	No
IX	Yes	Yes	No	No	No	No
S	Yes	No	Yes	Yes	No	No
U	Yes	No	Yes	No	No	No
SIX	Yes	No	No	No	No	No
X	No	No	No	No	No	No

PROBLEMS WITH LOCKS

- **Suspension**
  - An incompatible lock is already held by another transaction
  - The current process has to wait
- **Timeout**
  - The suspensions exceeds an interval
  - The process is terminated with messages and error codes
    - SQLCODE -911 or -913, SQLERRD3 00C9008E
- **Deadlock**
  - Two or more processes hold a lock that the other needs in order to release
  - Db2 selects a victim based on its characteristics, attempts to rollback, and terminates the process

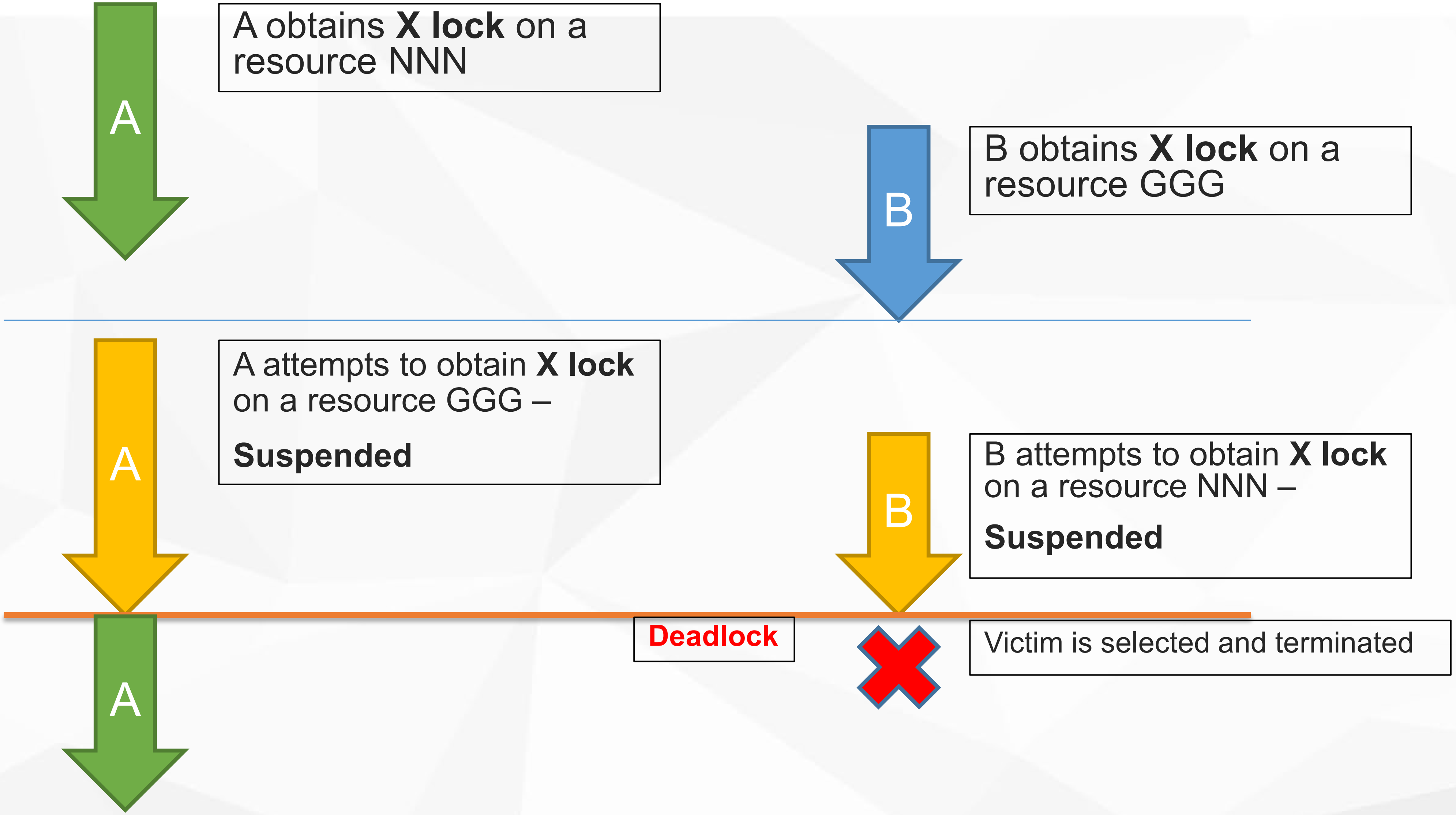
LOCK SUSPENSION AND TIMEOUT



B requests an incompatible lock on resource NNN

Timeout - B is terminated

DEADLOCK



A obtains **X lock** on a resource NNN

B obtains **X lock** on a resource GGG

A attempts to obtain **X lock** on a resource GGG –

Suspended

B attempts to obtain **X lock** on a resource NNN –

Suspended

Deadlock

Victim is selected and terminated

DB2'S CONTROLS - IRLMRWT AND DEADLOK

- **IRLMRWT** – Resource Timeout ZPARM
  - Lock request cannot be satisfied within the interval
  - Default is 30 seconds
  - **Online changeable** in Db2 13!
- **DEADLOK** – IRLM PROC Deadlock time
  - Amount of time for which deadlock detection cycles are to run
  - Default 1
  - 1-5 – seconds
  - 100 – 5000 – milliseconds

```
➢  MODIFY irlmproc,STATUS

DXR101I xxxx001 STATUS SCOPE=LOCAL
        DEADLOCK: 1000
SUBSYSTEMS IDENTIFIED
NAME       T/OUT    STATUS         UNITS        HELD     WAITING     RET_LKS
ssid        0060    UP                 6          19           0           0
    DXR101I End of display
```

TIMEOUT CALCULATIONS

- Wait time = timeout period * timeout multiplier
- Timeout period = Round(Resource Timeout / Deadlock time) * Deadlock time
  - Example: timeout period = 60/1 * 1 = **60**
- Timeout multiplier
  - Example: TSO: wait = 60*1 = 60, Utility = 60*6=360

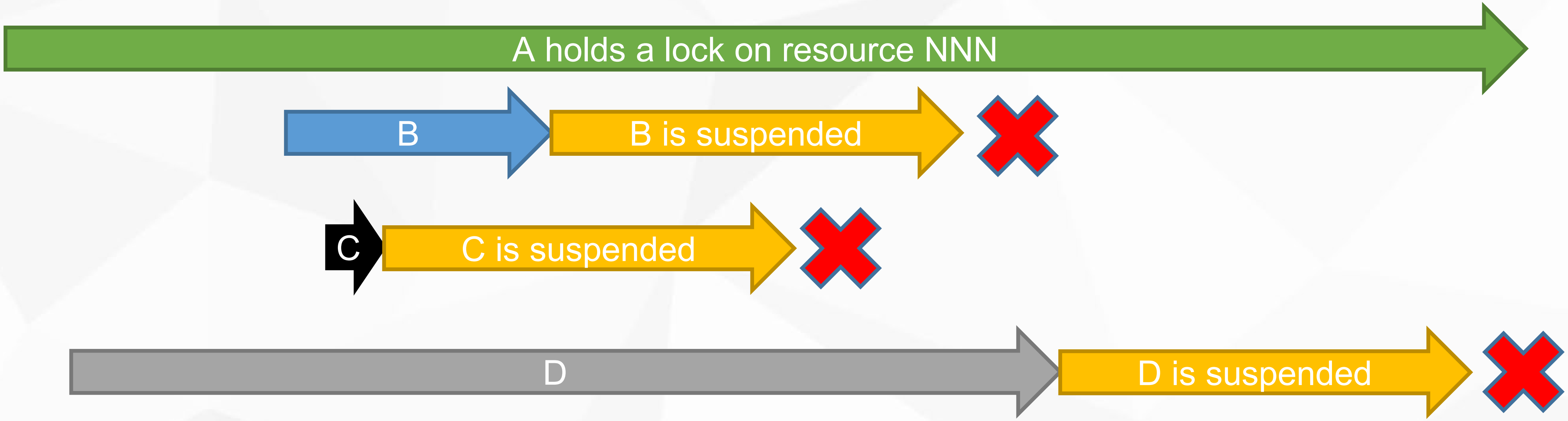
Type	Multiplier
IMS MPP, IMS FP, CICS, QMF, CAF, TSO, RRSAF, global txs	1
IMS BMPs	4
IMS DL/I batch	6
IMS Fast Path Non-message processing	6
BIND subcommand processing	3
STOP DATABASE command processing	10
Utilities	6 (UTIMOUT)
Retained locks	0

```
➢  MODIFY irlmproc,STATUS

DXR101I xxxx001 STATUS SCOPE=LOCAL
        DEADLOCK: 1000 (ms)
SUBSYSTEMS IDENTIFIED
NAME      T/OUT   STATUS          UNITS       HELD    WAITING    RET_LKS
ssid       0060   UP                  6         19          0          0
    DXR101I End of display
```

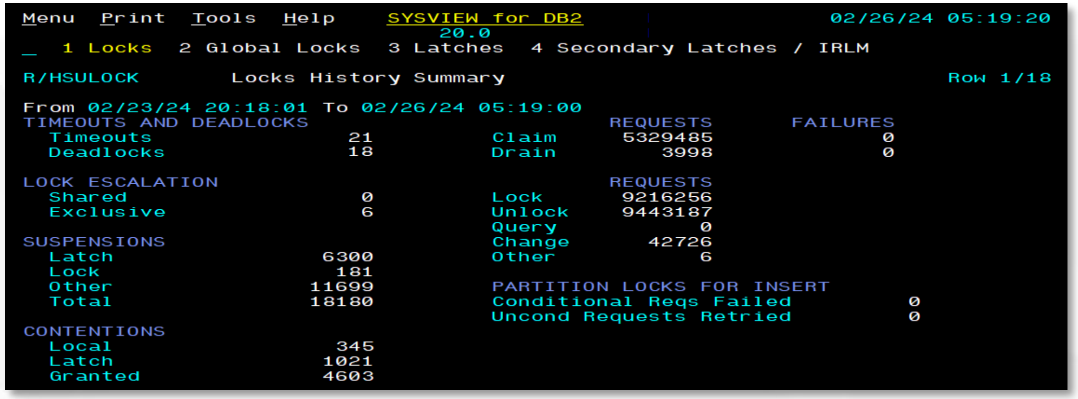
PROBLEM STATEMENT

- All applications are subject to the **same intervals**
- Different applications may have different needs
  - **Multi-tenancy**
  - Behavior from different platforms (**Db2 LUW**)



DB2 13 COMES TO RESCUE

- Applications can **modify** their **timeout** interval
- Is this for you? **Monitor** your applications!

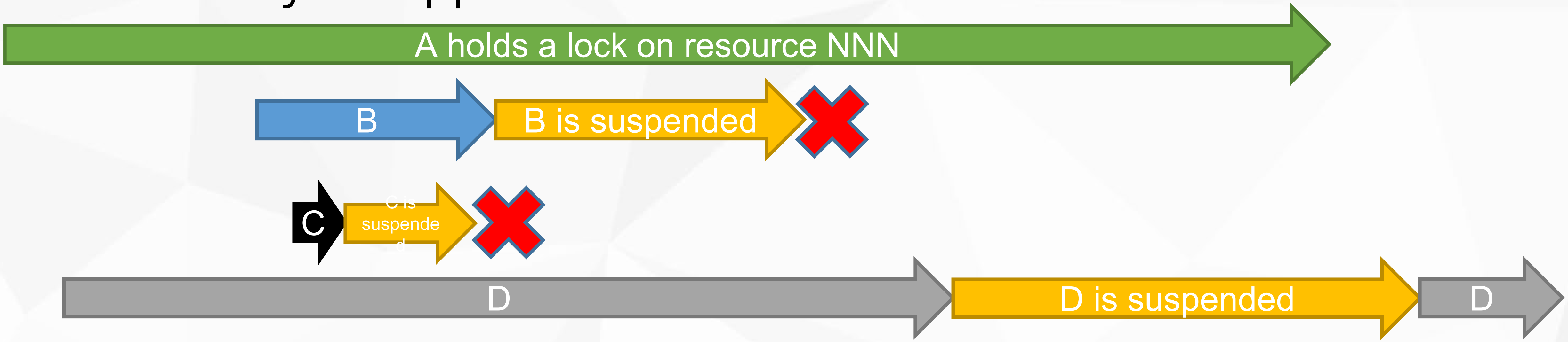


```
Menu  Print  Tools  Help     SYSVIEW for DB2                  02/26/24 05:19:20
                                  20.0
_  1 Locks  2 Global Locks  3 Latches  4 Secondary Latches / IRLM

R/HSULOCK        Locks History Summary                                Row 1/18

From 02/23/24 20:18:01 To 02/26/24 05:19:00
TIMEOUTS AND DEADLOCKS                         REQUESTS      FAILURES
  Timeouts                      21      Claim   5329485             0
  Deadlocks                     18      Drain      3998             0

LOCK ESCALATION                                REQUESTS
  Shared                         0      Lock    9216256
  Exclusive                      6      Unlock  9443187
                                        Query         0
SUSPENSIONS                             Change    42726
  Latch                       6300      Other         6
  Lock                         181
  Other                      11699      PARTITION LOCKS FOR INSERT
  Total                      18180      Conditional Reqs Failed       0
                                        Uncond Requests Retried       0
CONTENTIONS
  Local                        345
  Latch                       1021
  Granted                     4603
```



DB2 13 COMES TO RESCUE

- **Applications can modify their timeout interval?!?**
- V13R1M500 introduced **CURRENT LOCK TIMEOUT** special register
  - Number of seconds before a resource timeout is detected
  - **Overrides** the timeout value in IRLMRWT
    - The **default** comes from IRLMRWT
  - Does not affect P-lock processing, or processes related to plan and package allocation
  - Anyone can use the **special register!**
    - **No security control**
  - Comes from LUW

```
SELECT CURRENT LOCK TIMEOUT FROM SYSIBM.SYSDUMMY1;
  +----------------+
1_|             60 |
  +----------------+
```

DB2 13 COMES TO RESCUE

- Allowed values for **SET CURRENT LOCK TIMEOUT** :
  - `-1 | WAIT` - Wait indefinitely or to a deadlock
  - `0  | NOT WAIT` - Conditional lock
  - `NULL` - Reset to the default (IRLMRWT)
  - `(WAIT) 1-32767` – timeout seconds
  - Allows a variable also

```
                                           Result:
SET CURRENT LOCK TIMEOUT = 32767;          32767
SET CURRENT LOCK TIMEOUT = NULL;           60 (IRLMRWT)
SET CURRENT LOCK TIMEOUT = WAIT;           -1
SET CURRENT LOCK TIMEOUT = WAIT 1;         1
SET CURRENT LOCK TIMEOUT = NOT WAIT;       0
```

- The maximum allowed value is controlled by **SPREG_LOCK_TIMEOUT_MAX** ZPARM
  - Default is -1 - Any supported value can be specified
  - 0-32767 maximum value that can be specified
    - If SET CURRENT LOCK TIMEOUT is higher, then **SQLCODE -713, -725** (DRDA)

DB2 13 DEADLOCK VICTIM SELECTION

- Applications can set the deadlock priority
- New **DEADLOCK_RESOLUTION_PRIORITY** built-in **global variable**
  - Introduced with **V13R1M501**
  - Authorization is required – user needs **WRITE privilege** to set the value
    - Not granted to PUBLIC
  - Default is NULL
  - Value: 0-255 the higher value the less likely a deadlock victim

```
                                                             Result:
SELECT SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY FROM SYSIBM.SYSDUMMY1;  NULL
SET  SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY = 100;                    100
```

ONE MORE THING – APPLICATION GRANULARITY FOR LOCKING LIMITS

- Introduced with **V12R1M507**
- New built-in global variables
  - Require authorization!

Global Variable	Corresponding ZPARM	Applies to
SYSIBMADM.**MAX_LOCKS_PER_TABLESPACE**	NUMLKTS (default for LOCKMAX SYSTEM)	Max number of locks (page, row, LOB) the application can hold in **a table space**
SYSIBMADM.**MAX_LOCKS_PER_USER**	NUMLKUS	Max number of locks (page, row, LOB) the application can hold in **all table spaces**

PROFILE TABLES ENHANCEMENTS

PROFILE TABLES BENEFITS

- Can be used to set **global variables** and **special registers**
  - Added CURRENT LOCK TIMEOUT and DEADLOCK_RESOLUTION_PRIORITY
- **No need to update** the applications!
- Can now work even for **local applications** not only for remote!
- Plays well with the enhancements from the prior section

SUPPORT FOR LOCAL APPLICATIONS

- Criteria for identifying local application – **DSN_PROFILE_TABLE**:
  - AUTHID, ROLE
  - COLLID, PKGNAME
  - One of CLIENT_APPLNAME or CLIENT_USERID or CLIENT_WORKSTNNAME
- Specify the attributes – **DSN_PROFILE_ATTRIBUTES**
  - KEYWORDS = ‘SPECIAL_REGISTER’ or ‘GLOBAL_VARIABLE’
  - ATTRIBUTE1 = SET statement
  - ATTRIBUTE2 = NULL (remote only) or 1 (local only) or 2 (both)
  - ATTRIBUTE3 = NULL

EXAMPLE

- Local application with **AUTHID** USER01
- Set **CURRENT LOCK TIMEOUT** to 15
- Set **DEADLOCK_RESOLUTION_PRIORITY** to 5

```
INSERT INTO SYSIBM.DSN_PROFILE_TABLE(PROFILEID, AUTHID, PROFILE_ENABLED) VALUES
                                    (1,‘AUTH01','Y');
INSERT INTO SYSIBM.DSN_PROFILE_ATTRIBUTES(PROFILEID, KEYWORDS, ATTRIBUTE1,
                               ATTRIBUTE2) VALUES (1, 'SPECIAL_REGISTER', 'SET
                               CURRENT LOCK TIMEOUT = 15', 1);
INSERT INTO SYSIBM.DSN_PROFILE_ATTRIBUTES(PROFILEID, KEYWORDS, ATTRIBUTE1,
                               ATTRIBUTE2) VALUES (1, 'GLOBAL_VARIABLE‘,
                               'SET SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY = 5‘, 1);
```

PROFILEID	AUTHID
1	USER01

PROFILE ID	KEYWORDS	ATTRIBUTE1	ATTRIBUTE2	ATTRIBUTE3
1	SPECIAL_REGISTER	SET CURRENT LOCK TIMEOUT = 15	1	NULL
1	GLOBAL_VARIABLE	SET SYSIBMADM.DEADLOCK_RESOLUTION_PRIORITY = 5	1	NULL

PROFILE TABLES - CONSIDERATIONS

- **ZPARM DDF must be set to YES / AUTO** (even of we want to use it for local apps)
  - Otherwise DSNT761I and “DDF NOT LOADED” in DSN_PROFILE_ATTRIBUTES_HISTORY
- Proper **function level** is required
- Profile is scanned on the package load (first SQL in the package)
  - Reloaded after COMMIT/ROLLBACK if RELEASE(COMMIT)
- You cannot use host variables
- Package for an UDF or SP is always local even when called from a remote app
- Application SET takes a precedence over profile table
- Special registers and global variables are not saved/restored
  - Recommendation: **Do not mix app SET and profile tables**
- Same security requirements as if issued directly by an app

APPENDIX

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS

- Since the dawn of man... (Db2 for MVS Version 1 Release 1 in 1985)
  - we all know INSERT, UPDATE, DELETE, SELECT, COMMIT, ROLLBACK
- Db2 1.2 (1986)
  - EXPLAIN
    - use it whenever testing/investigating your SQL
- Db2 1.3 (1987)
  - DATE, TIME, TIMESTAMP data types support, date time arithmetic
  - check your PLAN_TABLE - column TIMESTAMP vs EXPLAIN_TIME (Db2 10)
  - UNION ALL
- Db2 2.1 (1988)
  - Referential integrity - Db2 managed RI vs application managed RI?
  - what about other constraints? (check, unique)

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 2.3 (1990)
  - Packages
    - DBRMs cannot be bound into plans since Db2 10
  - OPTIMIZE FOR n ROWS
- Db2 4 (1995)
  - Data Sharing (almost transparent to application developers)
  - External Stored Procedures
  - Outer joins
  - WITH UR clause
- Db2 5 (1997)
  - Result sets from stored procedures
  - Conformance to the SQL-92 standard
  - Do you remember Sysplex query parallelism? (Deprecated in Db2 10)

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 6 and Db2 6 refresh (1998)
  - Triggers
  - Large Objects (LOBs)
  - SQL Stored Procedures - converted to C
  - User Defined Functions
  - Distinct types - object relational extensions
  - SAVEPOINTs
  - Identity columns
  - Declared Temporary Tables

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 7 (2001)
  - Unicode support
  - Scrollable cursors
  - Limited fetch - FETCH FIRST n ROWS
  - Row expressions
  - Unions in Views
  - Compliance with SQL 99
  - ORDER BY expression
  - SQL Scalar functions (inline)
  - Stored procedures in Java

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 8 (2004)
  - (First version I’ve met)
  - Multi-row inserts and fetches
  - 64-bit support
  - Long names (128 characters instead of 18)
  - Informational referential constraints
  - Materialized Query Tables
  - Online Schema Evolution
  - INSERT within SELECT
  - Sequences
  - GET DIAGNOSTICS – ANSI/ISO 1999
  - Session variables
  - Common Table Expressions

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 9 (2007)
  - pureXML
  - New built-in timestamp functions (TIMESTAMPADD, TIMESTAMP_ISO)
  - Native SQL Stored procedures
  - INSTEAD OF Triggers
  - Clone tables
  - SKIP LOCKED DATA
  - Binary data types - are you still using VARCHAR FOR BIT DATA?
  - OLAP functions
  - EXCEPT/INTERSECT
  - Universal Tablespaces – PBR, PBG
  - SELECT FROM UPDATE and SELECT FROM DELETE
  - TRUNCATE, MERGE
  - Optimistic locking – ROW CHANGE TIMESTAMP

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 10 (Oct 2010)
  - Temporal tables
    - System, business, bi-temporal
  - Extended timestamp support - new precision, time zones
  - Access to CURRENTLY COMMITTED data
  - SELECT FROM table change reference – MERGE support added
  - Non-inline SQL scalar functions
  - SQL table functions
  - Implicit casting – SQL standard compliance
    - BIF compatibility
  - OLAP – moving sums and moving averages. RANK, DENSE_RANK, ROW_NUMBER
  - Column masks and row permissions

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 11 (2013)
  - Transparent archiving
  - Expanded support for temporal tables
  - Global variables
  - Support for SQL arrays
  - Improved SQL PL
  - Autonomous procedures
  - Enhancement to LIKE predicate
    - LIKE_BLANK_SIGNIFICANT ZPARM
  - (Expanded RBA/LRSN)

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- Db2 12 (2016)
  - Continuous delivery
  - Full MERGE
  - piece-wise DELETE
  - SQL pagination
  - Advanced triggers
  - Dynamic plan stability

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- V12R1M501 (May 2017)
  - new LISTAGG function, SQL 2016
- V12R1M502 (May 2018)
  - Support for more granular encrypted objects
  - New KEY LABEL option in DDL
  - Explicit casting of numeric data types to fixed or variable graphic strings
- V12R1M503 (Oct 2018)
  - Db2 AI for z/OS
  - replication of system-period temporal tables and generated expression columns
  - Temporal auditing
- V12R1M504 (April 2019)
  - z14 Huffman compression
  - Prevent creation of new deprecated objects
  - New passthrough built-in function for Analytics Accelerator
  - New SQL syntax alternatives

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- V12R1M505 (June 2019)
  - Improved Hybrid Transactional Analytical Processing performance
  - Rebind phase-in for packages that are being used for execution
  - New built-in functions for encryption and decryption with key labels
    - ENCRYPT_DATAKEY, DECRYPT_DATAKEY functions
  - Temporal and archive transparency support for WHEN clause on triggers
- V12R1M506 (Oct 2019)
  - Alternative function names support
- V12R1M507 (June 2020)
  - Application granularity for locking limits (NUMLKUS and NUMLKTS)
  - CREATE OR REPLACE for procedures
  - Newly supported passthrough-only expressions with IBM Db2 Analytics Accelerator

A BRIEF HISTORY OF SQL IN DB2 FOR Z/OS, CONT.

- V12R1M508 (Oct 2020)
  - Migration of multi-table table spaces to partition-by-growth universal table spaces (UTS)
- V12R1M509 (Feb 2021)
  - Support for tamper-proof audit policies
  - High availability for accelerator-only tables
  - Specify a compression algorithm at the table, table space, or partition level
  - Temporal RI allows UPDATE or DELETE on the parent table
- V12R1M510 (Apr 2021)
  - Enforces the completion of various pre-migration activities before migration to the next Db2 release
    - All Db2 12 function levels are activated.
    - All catalog updates for Db2 12 are applied, and the Db2 catalog is at the last Db2 12 catalog level.
    - All recently used application packages are rebound recently enough to ensure that they are supported by the next Db2 release.
    - packages that were used in the last 18 months and were last bound or rebound in a release before Db2 11
  - Does not introduce any new features or capabilities, or catalog changes
  - It is not a fallback SPE for migration