IDUG Leading the DB2 User Community since 1988
IDUG DB2 Tech Conference
Barcelona, Spain - October 2013

Analyzing DB2 for z/OS Resource Serialization and Concurrency Problems

Bart Steegmans
IBM

Session Code: A13
Wednesday 16 October 2013 15:45 - 16:45 | Platform: DB2 for z/OS

While Locking and other means of concurrency control are essential to any database management system, they are often the cause of problems in today's production systems.

Applications are not always designed with concurrency in mind, so it is not uncommon that with increased transaction volumes concurrency issues start to surface in your production systems.

People dealing with the production systems are not always involved in the design of the applications they manage, so they often have little knowledge of the application locking behavior. This is even more true for applications that are not developed in-house.

Therefore it is important to understand how to detect (potential) locking problems, and how to analyze them to find the root cause of the problem with little or no prior knowledge of the application.

Agenda

- Types of locking problems
  - Timeout
  - Deadlock
  - Lock escalation
  - Long (lock) suspension times
  - Excessive number of locks acquired
- How do I find out I have a problem ?
  - Concurrency warning signs
  - (Periodic) monitoring for concurrency problems
- Analyzing concurrency problems – The toolbox
  - DB2 commands and EXPLAIN
  - DB2 Traces
- Analysis of a simple deadlock scenario and solution

In this session, we discuss:

* Different types of locking and concurrency problems

* Concurrency warning signs

* How to set up periodic monitoring for concurrency problems

We also describe the different tools you have at your disposal to investigate a concurrency problem in order to discover the root cause of the problem. The toolbox consists of:

* DB2 commands

* EXPLAIN statement and the information in the dynamic statement cache

* Standard DB2 traces

* Detailed DB2 traces

* Using online monitor

Lastly we demonstrate how to perform this type of analysis for a simple deadlock scenario.

The session assumes that people are familiar with DB2 serialization mechanisms.

The target audience are application designers, DBAs and system administrators.

Acknowledgements and Disclaimers

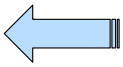
Availability. References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates.

The workshops, sessions and materials have been prepared by IBM or the session speakers and reflect their own views. They are provided for informational purposes only, and are neither intended to, nor shall have the effect of being, legal or other guidance or advice to any participant. While efforts were made to verify the completeness and accuracy of the information contained in this presentation, **it is provided AS-IS** without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this presentation or any other materials. Nothing contained in this presentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.

Many thanks to Gopal Krishnan and Ting Xu Guan for reviewing my foils.

DB2 Serialization Mechanisms

- **Transaction locking – managed by IRLM** 
  - Lock size - what is locked (row, page, table ...)
  - Lock state or lock mode - how is it locked (S(hared), (e)X(clusive), ..)
  - Lock duration - how long is it locked(commit, manual, ..)
  - Lock avoidance
- Claims and drains – managed by IRLM and DB2
- Latches – managed by DB2
  - BM page latching for index and data pages
  - DB2 internal latching (many latches grouped into 32 latch classes)
- Latches – managed by IRLM
  - Internal IRLM serialization

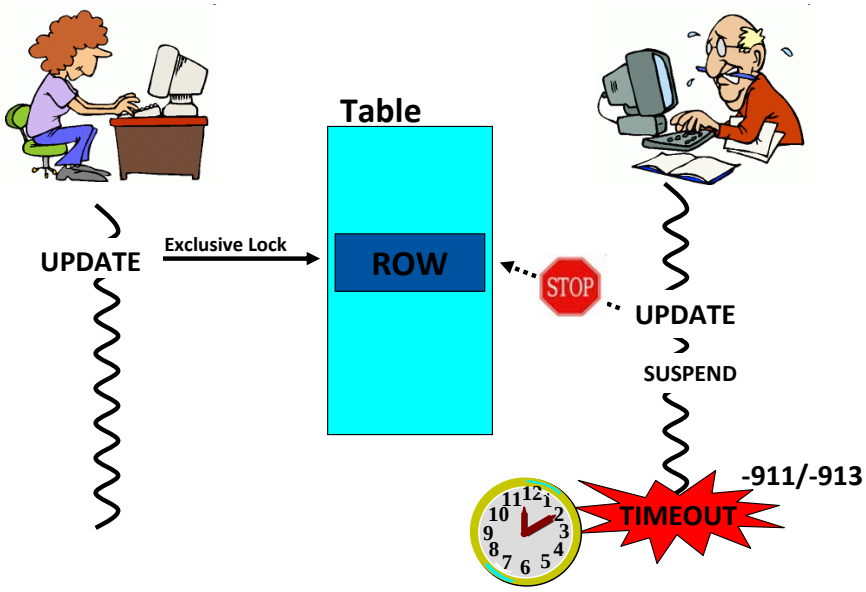
In this session we focus on IRLM transaction locking and issues with those.

There are many other serialization mechanisms used by DB2 but as the session is only one hour we have to limit the scope somewhat.

Types of locking problems

- Timeout
- Deadlock
- Lock escalation
- Long (lock) suspension times
- Excessive number of locks acquired

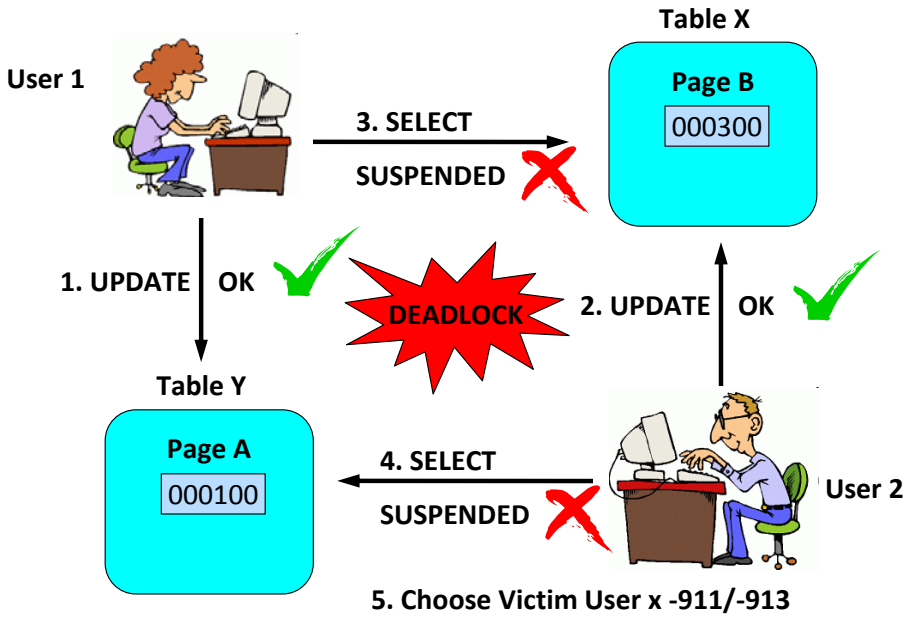
Timeout



When a transaction requests a lock on a resource in an incompatible state, it is suspended.

If the lock does not become available after the timeout interval (IRLMRWT) has expired, the thread receives a timeout condition.

Deadlock (with 2 resources)

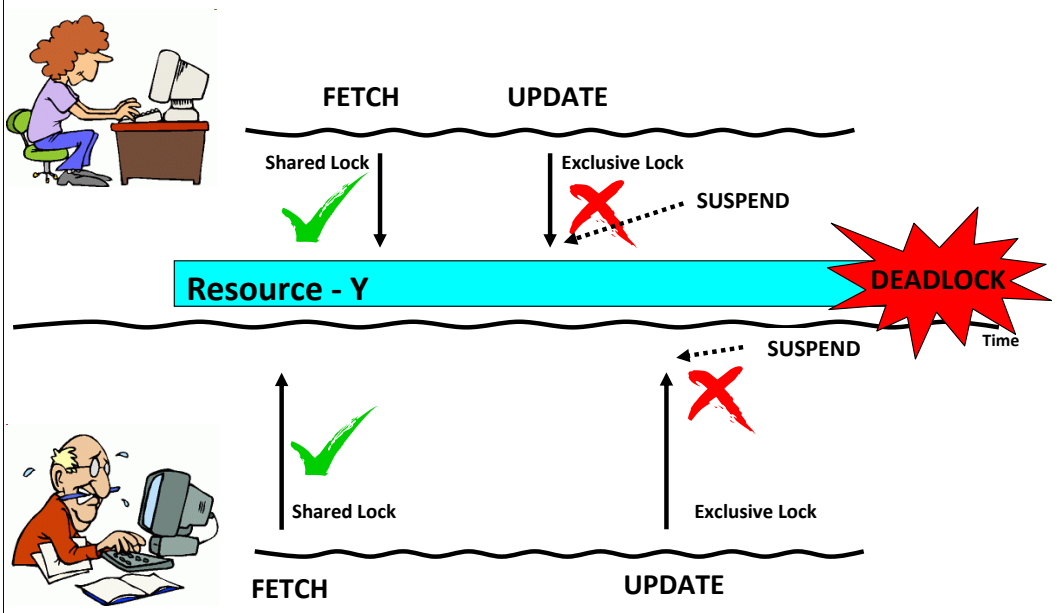


It is possible for thread to acquire locks in certain ways that they end up in a deadlock situation. each waiting on the other to release its locks but as they are all waiting they

will never continue to be able to release their locks.

IRLM has code to detect these deadlock conditions and presents them to DB2. Then DB2 decides which thread is selected at the deadlock 'victim' (receiving a negative SQLCODE).

Deadlock (with 1 resource)



Note that both FETCH statements are retrieving (and locking) the same resource.

The assumption is that the FETCH statements are required to acquire a S-lock (no lock avoidance for whatever reason).

Then the UPDATE statements update the same row that was previously fetched.

Even though the user already has an S-lock on the resource, it can not upgrade to an X-lock because there is another holder of an S-lock on the same resource at that time (and S and X are not compatible).

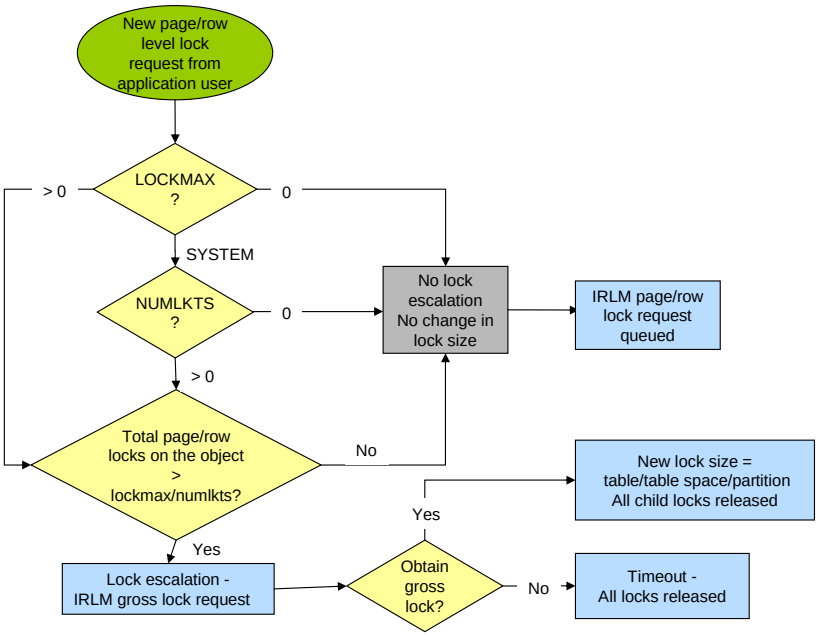
Lock Escalation

- "Protection mechanism" to avoid taking/managing too many page/row locks
- Replace all page/row locks by a gross lock (TB/TS/PART) of the same state
- Triggered by
  - LOCKMAX keyword on CREATE / ALTER TABLESPACE
  - ZPARM NUMLKTS (when using LOCKMAX SYSTEM)
- Disabled by using LOCKMAX 0 (zero)
- Lock escalation for partitioned TS
  - #locks is tracked at the TS level
  - Escalation only occurs on the partitions that have been accessed by the thread

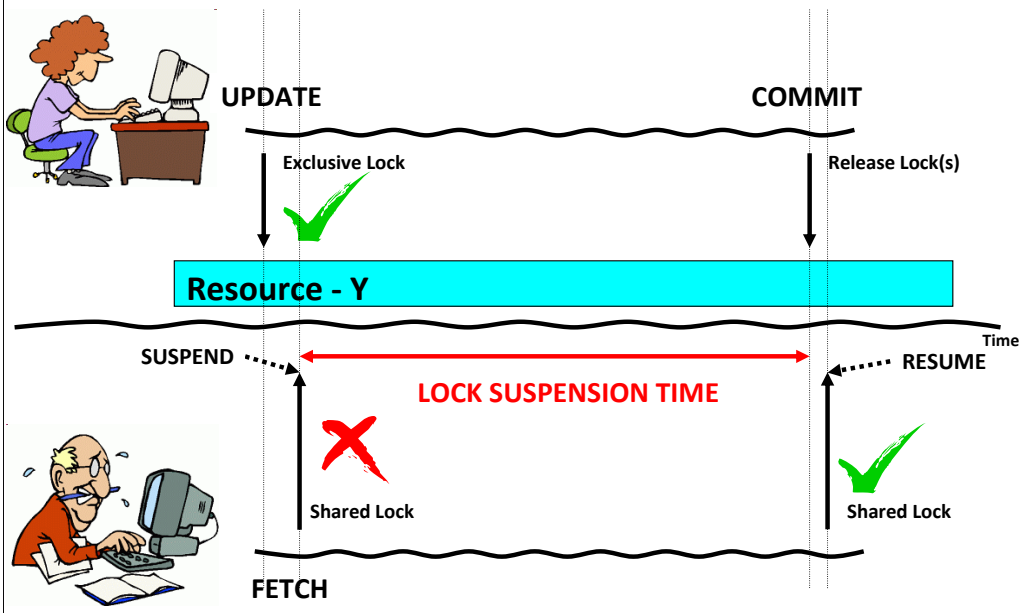
Lock escalation for partitioned TS used to be controlled by LOCKPART YES/NO setting.

Since V8 (and locking protocol 2) all partitioned TS are treated as if LOCKPART YES was specified.

Lock Escalation - 2



Long lock suspensions



A sample scenario will be discussed at the end of the presentation.

Excessive Number of Locks Acquired

- "There is no lock like no lock"
- Acquiring a lock takes CPU cycles
  - Charged to the user address space (when granted immediately)
- May or may not cause concurrency problems
  - With little concurrent work, may not be a problem
  - Problems can creep in when work and # concurrent threads increase over time
  - Increases the chances of lock suspensions
- Check whether
  - Isolation levels are set correctly
    - Use ISO(UR) when possible (but required good coding practices !)
    - Use ISO(CS) otherwise, try to avoide ISO(RS) and ISO(RR)
  - Lock avoidance is used
  - Commit frequency is ok

Using ISO(UR) is great for concurrency but requires that the application understands it is reading with UR and updates have to make sure they are actually checking that the value that was read is a 'valid' value (not a value that was 'in transition' but was never committed in the database)..

How do I find out I have a problem ?

- Concurrency warning signs
  - User warnings
  - **Console messages**
- (Periodic) monitoring for concurrency problems
  - **Which documentation to gather**
  - Analyzing DB2 statistics data
  - Analyzing DB2 accounting data
  - Monitoring DB2 system level concurrency related IFCIDs
  - (Exception monitoring using your favorite monitoring tool (like OMPE))

Concurrency Problems – Warning Signs - Console messages

- DSNT375I – Deadlock message
- DSNT376I – Timeout message
- DSNT501I – Resource unavailable msg
- DSNI031I – Lock escalation
- DSNJ031I – Uncommitted UR with many log records
  - Driven by URLGWTH ZPARM value
- DSNR035I – Uncommitted UR for a number of system checkpoints
  - Driven by URCHKTH ZPARM value
- DSNB260I – Long running reader
  - Driven by LRDRTHLD ZPARM value (message added in DB2 10)
- Make sure to monitor for these with your automation software

Prior to DB2 10 there there was no warning message for long running readers exceeding LRDRTHLD, only a trace record IFCID 313

Concurrency problems – Periodic monitoring - What data to gather ?

- DB2 statistics trace data – subsystem level info and 'exceptions'
  - Stats class 1 – System wide info about the work performed by the subsys
  - **Stats class 3 – Extremely valuable for concurrency analysis**
    - Deadlock (IFCID 172)
    - Timeout (IFCID 196)
    - Lock escalation (IFCID 337)
    - Long running UR (IFCID 313)
  - [Stats class 4 – DDF execption conditions]
  - [Stats class 5 – Data sharing statistics]
  - [Stats class 9 – Aggregate accounting statistics] (New in V10 - PM62797 Aug 2012)
  - Use STATIME = 1 – (very) low overhead

[ ] - good to activate all the time, but not relevant for concurrency issues

Stats class 9 (IFCID 369) only produces data when DB2 acctg trace is also active

Easy way to get a system wide overview of transaction behavior without having to run reports against all DB2 acctg records from that period

Note that starting with DB2 10 STATIME only applies to IFCIDs 0105, 0106, 0199, and 0365.
IFCIDs 0001, 0002, 0202, 0217, 0225, 0230 are always written at 1 minute intervals.

Concurrency problems – Periodic monitoring - What data to gather ? - 2

- DB2 accounting data – transaction level info
  - **Acctg class 1 – Total ET and CPU time of a plan/thread and many useful counters (including locking related counters)**
  - Acctg class 2 – ET and CPU inside DB2 is collected
    - DB2 has to sta/sto the clock each time application enters/exits DB2
    - More overhead than class 1 (typically 2.5% for OLTP – can be more for batch if they do many SQL operations (entry/exit) DB2)
    - Very valuable info – so best to turn on unless you really cannot afford it
  - Acctg class 3 – Allows to track known suspension events/times in DB2
    - Low overhead (except when very high # DB2 internal latch suspensions)
    - **Very valuable for concurrency analysis**
    - Suspension times include time waiting for a local lock, DB2 internal latch, global lock (data sharing)

Concurrency problems – Periodic monitoring - What data to gather ? - 3

- DB2 accounting data – transaction level info
  - Acctg class 1,2,3 – Plan level information
  - Accounting 7,8,10 – Package/DBRM level information
  - Acctg class 7 – ET and CPU at the package level (like class 2 for plans)
  - Acctg class 8 – Known suspension events/times in DB2 at the package level (like class 3 for plans)
  - Acctg class 10 – Package level detail info
    - Higher overhead
    - BP +SQL + Locking info at the package level
    - Locking info at package level can be very useful when analyzing concurrency problems

Concurrency problems – Periodic monitoring - What data to gather ? - 4

- DB2 Statistics class 1,3,4,5 [8,9]
  - Always !
  - Use STATIME=1
- DB2 Accounting class 1,2,3,7,8
  - Most people can afford it
  - Otherwise, use class 1, 2, 3
  - If that is even too much, use class 1,3
  - Always active (if you can afford it)
  - Otherwise turn it on for an hour during peak time every day
  - Use SMFCOMP=YES to reduce the volume rather than ACCUMACC
    - With ACCUMACC you lose transaction granularity which can make analyzing problems very cumbersome
- **No data means you need another occurrence before you can fix**

Analyzing Concurrency Problems – The Toolbox

- DB2 commands
- SQL EXPLAIN statement and DSC info
- "Standard" DB2 traces
  - DB2 Statistics
  - DB2 Accounting
- Detailed DB2 traces
- [Using online monitors]

The Toolbox – DB2 Commands

- -DISPLAY DATABASE
  - USE
  - RESTRICT
  - CLAIMERS
  - LOCKS
    - **Does not provide info about all locks (eg. page/row locks are not shown)**
  - Output contains the AGENT TOKEN that can be used to find the thread in the -DIS THD output
- -DISPLAY THREAD
  - TYPE(* / ACTIVE / INACTIVE / SYSTEM / INDOUBT / POSTPONED / PROC)
    - Note that '*' displays only active, indoubt, postponed, and system threads
  - Use LIMIT(*) to avoid output truncation

The Toolbox – DB2 Commands – DIS DB … LOCKS

- Interpreting -DIS DATABASE LOCKS output

```
DSNT397I  -DB9A
NAME     TYPE PART  STATUS           CONNID   CORRID       LOCKINFO
-------- ---- ----- ---------------- -------- ------------ ---------
TSBART2  TS         RW               TSO      BART         H-IX,S,C
   -                   AGENT TOKEN 10737
```

Status	State		Type	Duration
	L-lock + drain lock	P-lock		
H: Holding	IS	IX	S: TS L-lock	H: A: Allocation C: Commit H: Close cursor M: Freed by system P: Plan complete I: Interest (PS P-lock)
W; Waiting	IX	IX	T: TB L-lock	
R: Retained	S	S	P: PT L-lock	
	U[ab]		C: CS drain lock	
	SIX[b]	SIX	R; RR drain lock	
	X	X	W: WR drain lock	W: Number in the waiting queue
		NSU	PP: PS P-lock	R; N/A

a. Not applicable to LOBs
b. Not applicable to drain locks

The Toolbox – SQL EXPLAIN statement

- PLAN_TABLE - TSLOCKMODE column
  - Indication of the lock that will be acquired by the SQL stmt at execution time
  - Info depends on whether or not DB2 can determine the isolation level at bind/prepare time
  - Only applies to gross L- locks (TS/table lock)
    - Not page/row locks
    - Not index lock as there typically are no lock on the index – DB2 uses “data-only locking” since the introduction of T2 indexes in V4
    - Not data sharing P-locks

The Toolbox – Dynamic Statement Cache info

- DSC info is easily retrieved via **EXPLAIN STMTCACHE ALL** into **DSN_STATEMENT_CACHE_TABLE** table
- Contains statement level info about cached dynamic SQL about
  - Columns listed are limited to info relevant to concurrency issues
  - **STMT_ID** - StaTeMenT ID - EDM unique token for the statement
    - Can be used to correlate to info in msgs and IFCIDs
  - **STMT_TEXT** - Statement text of the SQL statement
  - **BIND_ISO** - Isolation BIND option that is in effect
  - **BIND_CDATA** - CURRENTDATA BIND option in effect
  - **STAT_SUS_LOCK** - Accumulated wait time for lock requests
  - **STAT_SUS_GLCK** - Accumulated wait time for global locks
  - **STAT_SUS_LATCH** - Accumulated wait time for latch requests
  - **STAT_SUS_PLATCH** - Accumulated wait time for page latch requests
  - **STAT_SUS_DRAIN** - Accumulated wait time for drain lock requests
  - **STAT_SUS_CLAIM** - Accumulated wait time for claim count requests

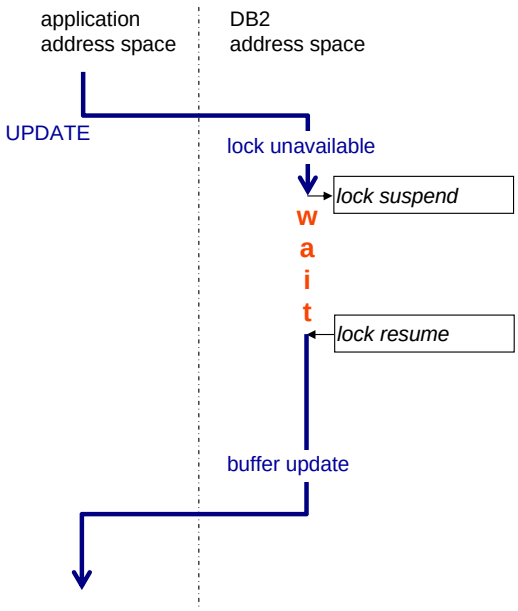
The Toolbox – Standard DB2 traces - Statistics

- Highlights , Subsystem Services , DRDA remote locations sections
- Non data sharing locking
- Data sharing locking section
- System address spaces CPU times
- Latch contention section
- [DML/DDL/DCL section]
- [GBP section]

The Toolbox – Standard DB2 traces - Accounting

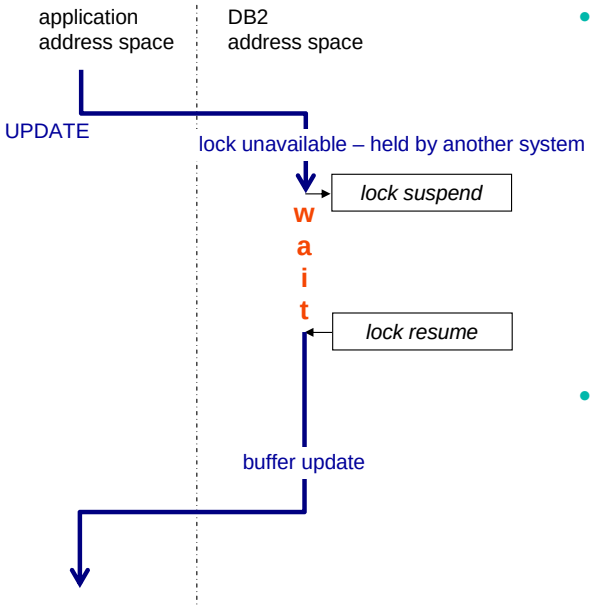
- Highlights section
- CLASS 1,2,3 (7,8) info
  - Many class 3 suspension counters related to concurrency
- Non data sharing locking
- Data sharing locking section
- Claim / Drain section
- [DML/DDL/DCL section]
- [GBP section]

The Toolbox – Standard DB2 traces - Accounting



- Thread can be suspended for:
  - Lock contention
  - IRLM latch contention
  - DB2 internal latch contention (Separate counter in V10)
- Page latch contentions are reported in a separate counter
- DB2 reports:
  - Accumulated time thread was suspended
  - Total number of suspensions

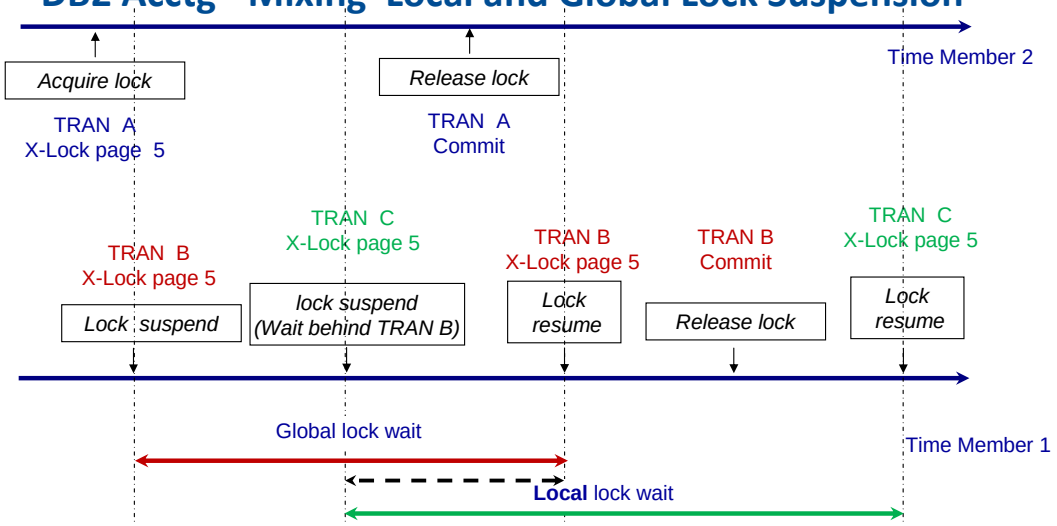
The Toolbox – Accounting Trace – Global Lock Suspension



- Thread can be suspended for:
  - Global L-parent lock
  - Global L-child (row/page) lock
  - Other global L-lock
  - Global page set P-lock
  - Global page P-lock
    - Data
    - IX leaf pages
    - SM pages
  - Other P-locks
- DB2 reports:
  - Accumulated time spent suspended
  - Total number of suspensions
  - Sum of all IRLM, XES, and fFalse contentions in Data Sharing Locking section

Page P-locks are managed by DB2 BM – More details about the type (data pages, SM pages, IX leaf pages) can be found in the GBP section of the DB2 accounting report

DB2 Acctg - Mixing Local and Global Lock Suspension



- Even though part of the wait time for tran C it is waiting for a global lock held by tran A, all suspension time is reported as local because at resume time, the lock that was released was local to this member

The Toolbox – Standard DB2 traces – stats class 3 IFCIDs

- IFCID 172 – deadlock info
- IFCID 196 – timeout info
- IFCID 313 – long running URs
- IFCID 337 – lock escalation

The Toolbox – Detailed DB2 traces - IFCIDs

- **Lock suspension information**
  - IFCID 44 - 45: IRLM lock suspend and lock resume request
  - IFCID 213 - 214: Start/end waiting for a drain lock
  - IFCID 215 - 216: Start/end waiting for the claim count to go to zero
  - IFCID 226 - 227: Start/end page latch wait (not an IRLM lock)
  - [IFCID 105, 107: Allow OBID translation from OBID numbers to object names]
  - You can use the following START TRACE command to gather this information:
    - -START TRACE(P) CLASS(30) TDATA(CPU COR DIST) DEST(SMF/GTF) IFCID(44,45,105,107,213,214,215,216,226,227)
- Allows you to create locking suspension reports (and traces)
  - Resources being suspended for
  - Threads being suspended

The Toolbox – Detailed DB2 traces - IFCIDs

- **Detailed locking information**
  - Lots of details - lots of records - [lots of] overhead
  - Non-data sharing
    - IFCID 21 - Detailed lock information - Traces each lock, unlock, change request sent to IRLM
    - IFCID 211 - Acquiring or releasing a claim
    - IFCID 212 - Acquiring or releasing a drain
  - Data sharing
    - IFCD 251 - Detailed information about page set P-lock requests.
    - IFCID 257 - IRLM notify processing
    - IFCID 259 - Traces information about page P-lock requests
  - Trace with IFCIDs to capture detailed locking and suspension info
    - Typically trace the suspensions also (see previous foils)
    - -START TRACE(P) CLASS(30) TDATA(CPU COR DIST) DEST(SMF/GTF) IFCID(251,257,259,21,211,212,44,45,213,214,215,216,226,227)

Tracing IFCID 251 can be a good alternative to tracing IFCID 21 if you are only interested in page set P-lock requests.

The processing impact of collecting IFCID 251 will be much less than when collecting IFCID 21.

It is a good idea to add IFCID 257 to the trace request, as it can give you a better understanding of the notify messages being sent between the

members of a data sharing group .

Tracing IFCID 259 can be a good alternative to tracing IFCID 21 if you are only interested in page P-lock requests.

Collecting IFCID 259 will have less of an impact on processing than collecting IFCID 21.

The Toolbox – Detailed DB2 traces - IFCIDs

- Information about lock avoidance
  - High level guesstimate via #UNLOCK / commit
  - IFCID 218 - Lock avoidance summary information
  - IFCID 223 - Detailed information about (successful use of) lock avoidance techniques
    - Beware of the trace volume/overhead
  - You can use the following START TRACE command to gather this information:
    - -START TRACE(P) CLASS(30) IFCID(218,223) TDATA(CPU COR DIST) DEST(SMF/GTF)

IFCID 218 indicates whether lock avoidance techniques have been used by a unit of work in general,

and for each object (page set), whether or not lock avoidance was used.

If IFCID 223 is active, a trace record is written each time lock avoidance is use. If lock avoidance was used on 100 pages, 100 trace records are produced.

If lock avoidance is doing a good job, tracing this IFCID can produce a very large number of trace records and will impact the CPU time of the transaction.

The Toolbox – Detailed DB2 traces - IFCIDs

- Information about DB2 internal latch suspensions
  - IFCID 51-52 - Shared latch resume and shared latch wait
  - IFCID 56-57 - Exclusive latch wait and exclusive latch resume
  - The following trace command will gather this information:
    - -STA TRA(P) CLASS(30) IFCID(56,57)
      DEST(SMF/GTF) TDATA(CPU COR DIST TRA)
  - Note that ther is is no IFCID capture an individual DB2 internal latch request
    - Only when a tran is suspended for a latch these records are cut
  - DB2 internal latches are managed by DB2 , not IRLM

These IFCIDs have the real DB2 latch number and the address of the latch. If the system experiences a high degree of DB2 latch contention, and if you want to identify the actual

latches that are triggering the latch contention, it is normally sufficient to trace only IFCID 56 and 57.

As there are many latch contentions, there must be X latch requests that will be suspended (since S and S are compatible), so in order to reduce the amount of trace data

being gathered, there is normally no need to trace IFCID 51 and 52.

The Toolbox – Detailed DB2 traces - IFCIDs

- Information about DB2 page latch suspensions
  - Managed by DB2 Buffer Manager component, not IRLM
  - IFCID 226 - 227: Start/end page latch wait
  - The following trace command will gather this information:
    - -STA TRA(P) CLASS(30) IFCID(226,227)
      DEST(SMF/GTF) TDATA(CPU COR DIST)
  - Note that there is no IFCID to capture an individual page latch request (only when a tran is suspended for a page latch these are cut)
- Other IFCIDs to trace
  - To better understand the context it is recommended to also start an “SQL trace”
    - -STA TRA(P) CLASS(1,2,3) DEST(SMF/GTF) TDATA(CPU COR DIST)
  - If you also want the ‘scan’ activity, use
    - -STA TRA(P) CLASS(1,2,3,8) DEST(SMF/GTF) TDATA(CPU COR DIST)

Analyzing Concurrency Problems – Analysis of a Simple Deadlock Scenario

- Analyzing the joblog and syslog
- Analyzing the deadlock record
- Checking the DB2 accounting data
- Zooming in using detailed trace data
- The complete deadlock picture
- Fixing the problem

Now we will look at a deadlock problem in more detail.

It will show a way to approach the analysis of locking problems.

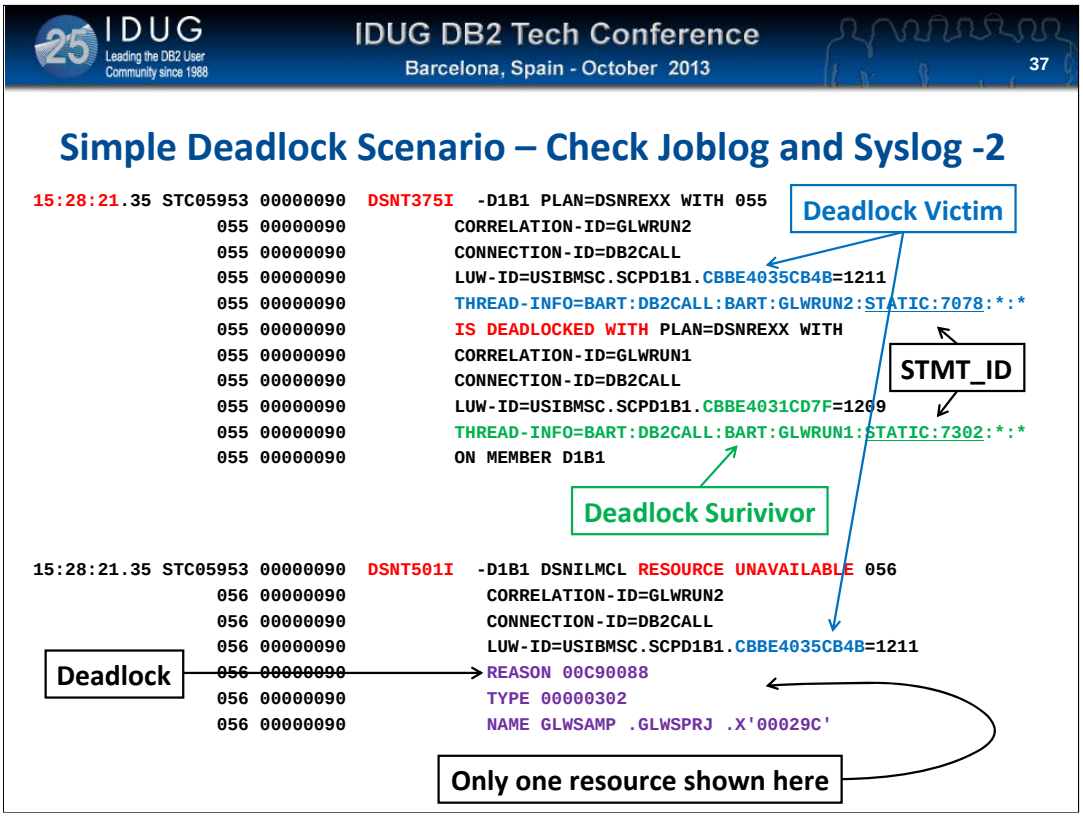
Simple Deadlock Scenario – Check Joblog and Syslog

- Check job output log for -911/-913
  - SQLERRD(3) also contains the reason code which indicates whether a deadlock or timeout occurred
    - 00C90088 – Deadlock
    - 00C9008E – Timeout

```
GLWR255I: Execution of Stored Procedures; Time: 15:27:59
    1 EMPADX   15:27:59 ; Objno:    1764 ; Tran:   43949 ; SQLrc: 0
  ... ......   ........ . ......   ..... . .....   ..... . ........
  127 EMPADD   15:28:20 ; Objno:   30706 ; Tran:   44222 ; SQLrc: 0
  128 EMPADD   15:28:20 ; Objno:   30707 ; Tran:   44223 ; SQLrc: 0
  129 DPTDEL   15:28:21 ; Objno:       0 ; Tran:   44224 ; SQLrc: -913
  130 EMPADD   15:28:21 ; Objno:   30708 ; Tran:   44225 ; SQLrc: 0
  131 EMPADD   15:28:21 ; Objno:   30709 ; Tran:   44226 ; SQLrc: 0
```

- Check syslog for DSNT375I and DSNT501I msg
  - Deadlock **victim** puts out DSNT375I and DSNT501I msg
  - See next foil

The scenario is driven by Mike Bracey's GLW tool



DB2 Codes manual – Appendix F – Resource types

Type = 00000302 -> Table space page -> format= DB.SP.PG

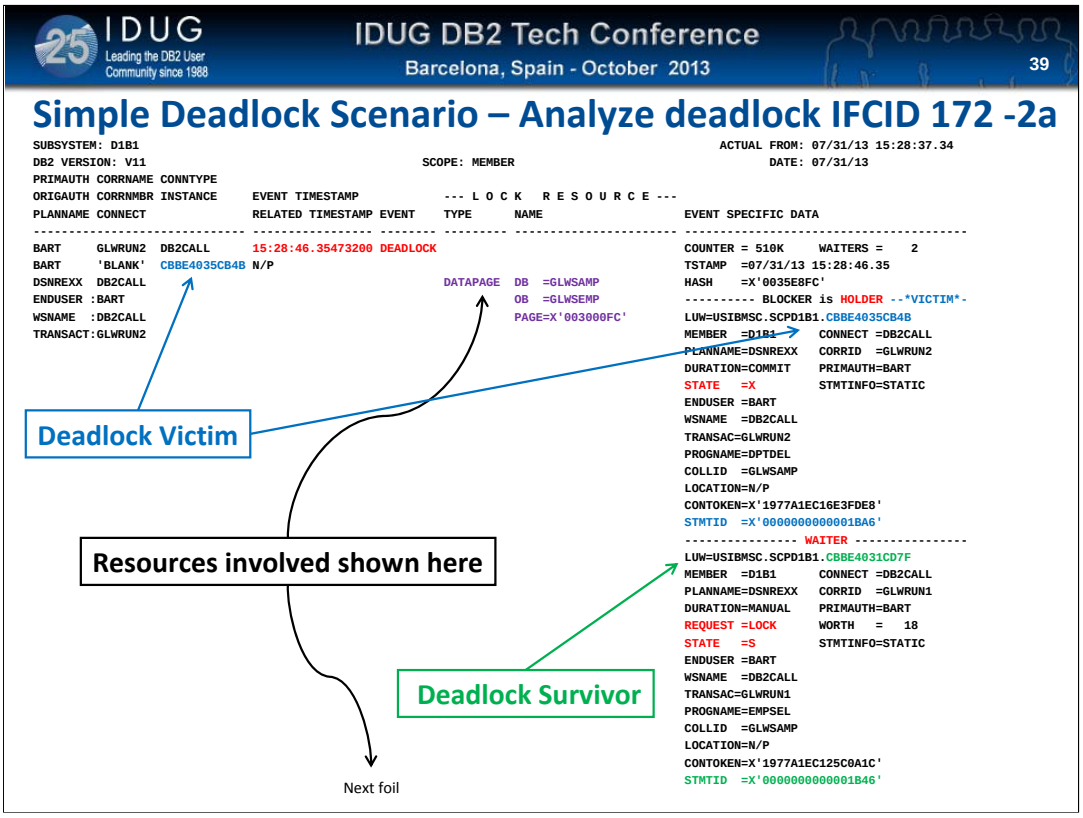
Simple Deadlock Scenario – Analyze deadlock IFCID 172

- #Resources involved , lock states involved, transactions involved, SQL statement (ID) involved
- IFCID 172 (and 196) originally had ALL holders and waiters involved in a deadlock or timeout
  - In busy systems, at any point in time IRLM is managing a very large number of locks
  - Finding all parties involved with their locks would be too resource intensive
  - But even if not all waiters are listed, the info should be enough to figure out the nature of the problem

(OMPE = OMegamon Performance Expert)

OMPE command to generate a lock trace report for all 'lockouts' (deadlocks and timeouts)

```
//SYSIN    DD  *
DB2PM
* ********************************
*      GLOBAL PARMS
* ********************************
 GLOBAL
      TIMEZONE (+4)
      INCLUDE( DB2ID(D1B1))
* ********************************
*      LOCKING REPORTS
* ********************************
 LOCKING
         TRACE
          LEVEL(LOCKOUT)
EXEC
```



Note that the time in IFCID 172 (deadlock) - **15:28:46.35473200 -** is not the same than the time of the DSNT375I message on the syslog **15:28:21.35**.

There is a 25 seconds difference – This is because of the system clock setup is taking LEAP seconds into account.

(Unfortunately OMPE can only correct for time zone differences and not leap seconds) so keep that in mind.

Simple Deadlock Scenario – Analyze deadlock IFCID 172 -2b

```
SUBSYSTEM: D1B1                                                    ACTUAL FROM: 07/31/13 15:28:37.34
DB2 VERSION: V11                          SCOPE: MEMBER                   DATE: 07/31/13
PRIMAUTH CORRNAME CONNTYPE
ORIGAUTH CORRNMBR INSTANCE     EVENT TIMESTAMP           --- L O C K   R E S O U R C E ---
PLANNAME CONNECT               RELATED TIMESTAMP EVENT   TYPE      NAME                  EVENT SPECIFIC DATA
------------------------------ ----------------- -------- --------- --------------------- ------------------------------------
BART     GLWRUN2  DB2CALL                                 DATAPAGE  DB  =GLWSAMP         HASH    =X'00AAAA9C'
BART     'BLANK'  CBBE4035CB4B                                      OB  =39               ---------- BLOCKER is HOLDER ----------
DSNREXX  DB2CALL                                                    PAGE=X'00029C'        LUW=USIBMSC.SCPD1B1.CBBE4031CD7F
ENDUSER :BART                                                                             MEMBER  =D1B1      CONNECT =DB2CALL
WSNAME  :DB2CALL                                                                          PLANNAME=DSNREXX   CORRID  =GLWRUN1
TRANSACT:GLWRUN2                                                                          DURATION=COMMIT    PRIMAUTH=BART
                                                                                          STATE   =X         STMTINFO=STATIC
                                                                                          ENDUSER =BART
                                                                                          WSNAME  =DB2CALL
                                                                                          TRANSAC=GLWRUN1
                                                                                          PROGNAME=EMPSEL
                                                                                          COLLID  =GLWSAMP
                                                                                          LOCATION=N/P
                                                                                          CONTOKEN=X'1977A1EC125C0A1C'
                                                                                          STMTID  =X'0000000000001C86'
                                                                                          ---------- WAITER -------*VICTIM*-
                                                                                          LUW=USIBMSC.SCPD1B1.CBBE4035CB4B
                                                                                          MEMBER  =D1B1      CONNECT =DB2CALL
                                                                                          PLANNAME=DSNREXX   CORRID  =GLWRUN2
                                                                                          DURATION=MANUAL    PRIMAUTH=BART
                                                                                          REQUEST =LOCK      WORTH   =   17
                                                                                          STATE   =U         STMTINFO=STATIC
                                                                                          ENDUSER =BART
                                                                                          WSNAME  =DB2CALL
                                                                                          TRANSAC=GLWRUN2
                                                                                          PROGNAME=DPTDEL
                                                                                          COLLID  =GLWSAMP
                                                                                          LOCATION=N/P
                                                                                          CONTOKEN=X'1977A1EC16E3FDE8'
                                                                                          STMTID  =X'0000000000001BA6'
```

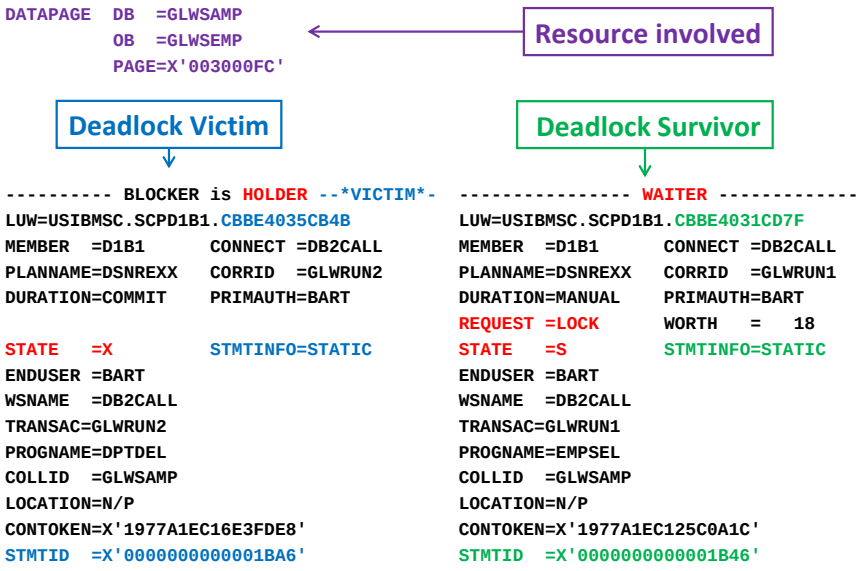
Deadlock Victim

Deadlock Survivor

Resources involved shown here

Prev foil

Simple Deadlock Scenario – Analyze deadlock IFCID 172 -3



This is the same as foil 2a before, with just the info about one resource and its holder and waiter

Simple Deadlock Scenario – Analyze deadlock IFCID 172 -4

```
--- L O C K   R E S O U R C E ---
TYPE      NAME
--------- ----------------------
DATAPAGE  DB  =GLWSAMP
          OB  =39
          PAGE=X'00029C'
```

Resource involved

Deadlock Survivor

Deadlock Victim

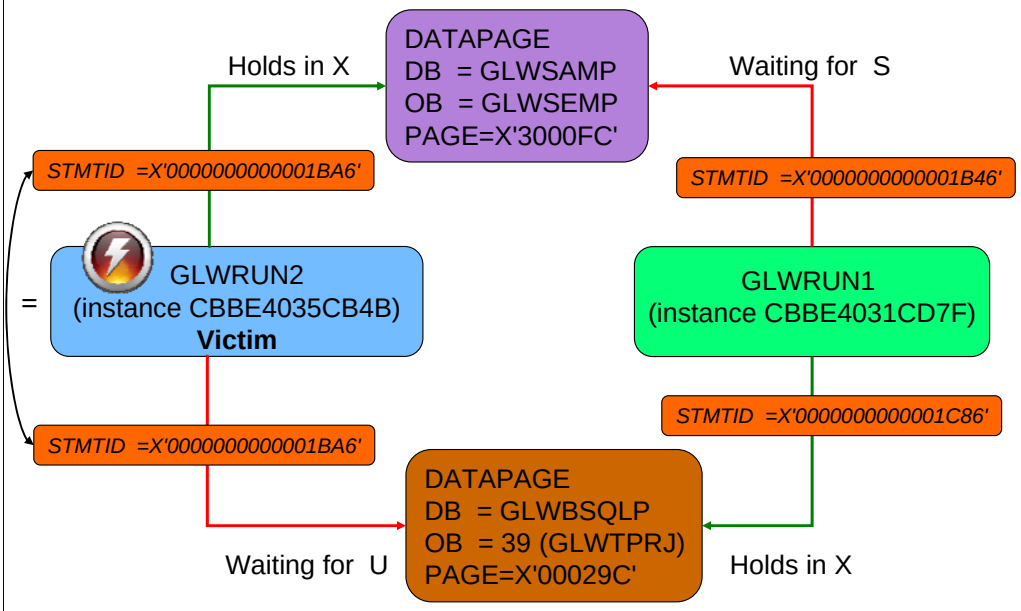
```
---------- BLOCKER is HOLDER ----------
LUW=USIBMSC.SCPD1B1.CBBE4031CD7F
MEMBER  =D1B1      CONNECT =DB2CALL
PLANNAME=DSNREXX   CORRID  =GLWRUN1
DURATION=COMMIT    PRIMAUTH=BART

STATE   =X         STMTINFO=STATIC
ENDUSER =BART
WSNAME  =DB2CALL
TRANSAC=GLWRUN1
PROGNAME=EMPSEL
COLLID  =GLWSAMP
LOCATION=N/P
CONTOKEN=X'1977A1EC125C0A1C'
STMTID  =X'0000000000001C86'

---------- WAITER -------*VICTIM*-
LUW=USIBMSC.SCPD1B1.CBBE4035CB4B
MEMBER  =D1B1      CONNECT =DB2CALL
PLANNAME=DSNREXX   CORRID  =GLWRUN2
DURATION=MANUAL    PRIMAUTH=BART
REQUEST =LOCK      WORTH   =   17
STATE   =U         STMTINFO=STATIC
ENDUSER =BART
WSNAME  =DB2CALL
TRANSAC=GLWRUN2
PROGNAME=DPTDEL
COLLID  =GLWSAMP
LOCATION=N/P
CONTOKEN=X'1977A1EC16E3FDE8'
STMTID  =X'0000000000001BA6'
```

This is the same as foil 2b before, with just the info about one resource and its holder and waiter

Simple Deadlock Scenario – What do we know so far ?



1) The resouce(s) involved
2) The locks being held and being waited on
3) The threads involved
4) The statements (ID) involved

Note that for the Victim there is only a single SQL statement (ID) involved.

Simple Deadlock Scenario – Find SQL Stmts Involved

- In DB2 10 deadlock detection tries to provide the STMT_ID of the SQL statement(s) involved and an indication whether the STMT_ID is for a static or dynamic SQL statement
  - STATIC – look in SYSPACKSTMT (our case)
  - DYNAMIC – look in the DSC (via EXPLAIN STMT_CACHE ALL)

```
SELECT  SUBSTR(COLLID,1,18) AS COLL_ID
     , SUBSTR(NAME,1,8)    AS PACKAGE_ID
     , QUERYNO
     , STMT_ID
     , HEX(STMT_ID) AS HEX_STMT_ID
     , STATEMENT
FROM SYSIBM.SYSPACKSTMT
WHERE HEX(STMT_ID)  IN (  '0000000000001BA6'
                        , '0000000000001B46'
                        , '0000000000001C86' );
```

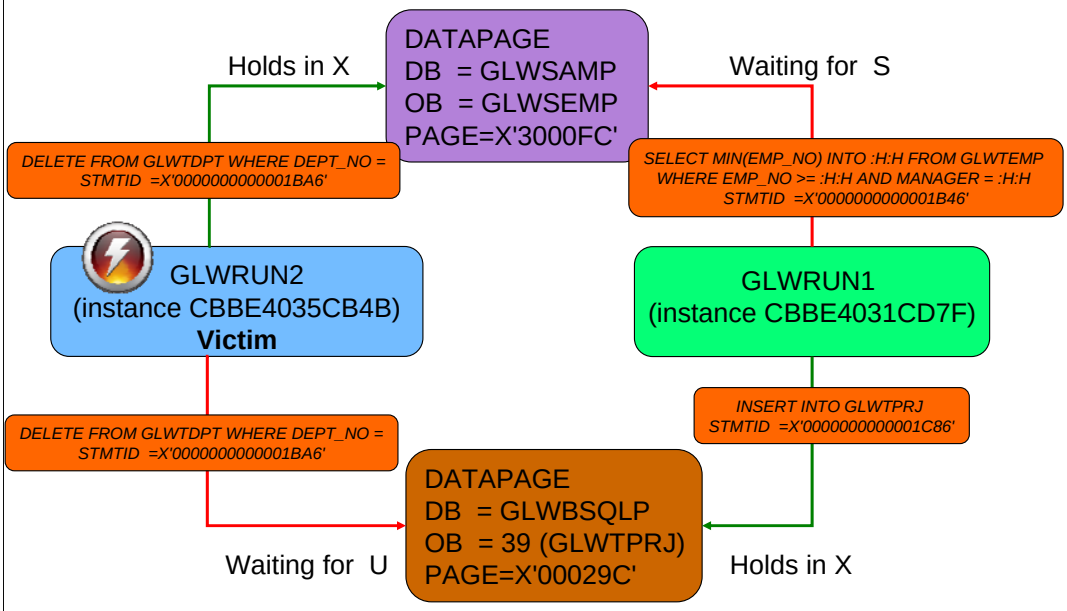
From the IFCID 172 record

Simple Deadlock Scenario – STMT_ID info

```
COLL_ID   PACKAGE_ID  QUERYNO   STMT_ID  HEX_STMT_ID
--------+---------+-----+----+---------+---------+------
GLWSAMP   EMPSEL         2610      6982  0000000000001B46   …
GLWSAMP   DPTDEL         1515      7078  0000000000001BA6   …
GLWSAMP   PRJADD            1      7302  0000000000001C86   …
```


```
STATEMENT
-+---------+---------+---------+---------+---------+---------+---------+---------+----------
SELECT MIN(EMP_NO) INTO :H:H FROM GLWTEMP WHERE EMP_NO >= :H:H AND MANAGER = :H:H QUERYNO 2610
DELETE FROM GLWTDPT WHERE DEPT_NO = :H:H QUERYNO 1515
INSERT INTO GLWTPRJ (PROJ_NO, PROJNAME, DEPT_NO, RESP_EMP_NO, PRSTDATE, PRENDATE ,PROJ_DESC, -
  CREATED_BY, UPDATED_BY )VALUES ( :H:H , :H:H, :H:H, :H:H, :H:H , :H:H, :H:H , USER , USER )
```

Simple Deadlock Scenario – What do we know so far ?



Use

SELECT * FROM SYSIBM.SYSTABLES WHERE OBID = 39 AND DBNAME = 'GLWSAMP' ;

to find out that OBID d'39' is tables GLWTPRJ

Simple Deadlock Scenario – DELETE accessing multiple tables ?

- So why is DELETE FROM GLWTDPT WHERE DEPT_NO = :H accessing
  - GLWSEMP, holding an X-lock, and
  - GLWTPRJ , trying to get a U-lock
- The answer lies in the DDL – DB2 Referential integrity:
  - ALTER TABLE GLWSAMP.GLWTEMP
    FOREIGN KEY GLWFEMP1 (WORKDEPT)
    REFERENCES GLWSAMP.GLWTDPT (DEPT_NO)
    ON DELETE SET NULL ;
  - ALTER TABLE GLWSAMP.GLWTPRJ
    FOREIGN KEY GLWFPRJ1 (DEPT_NO)
    REFERENCES GLWSAMP.GLWTDPT (DEPT_NO)
    ON DELETE CASCADE ;
- [Triggers can also have a similar effect]

Deleting rows from GLWTDPT results in setting the rows with a matching WORKDEPT col value in GLWTEMP to null.

Deleting rows from GLWTDPT results in deleting rows a matching DEPT_NO column value in GLWTPRJ (if the department is deleted all the project is has/had, are also deleted (DELETE CASCADE))

Simple Deadlock Scenario – Check the DB2 accounting record

- Run an accounting trace report of the transactions involved
  - Victim should show #deadlocks > 0
  - If you have package level detail info (acctg class 10) , the package locking section should allow to identify the package that expererienced the deadlock
    - There is less need for this in V10 as the STMT_ID should allow to identify the SQL statement and you can find the package that way too
  - The survivor will only show high lock suspension time
    - It is resumed 'normally' after being suspended until the deadlock got detected and broken by IRLM and DB2
    - No different than a case where the other thread had released the lock when it is done (for example as the result of executing its commit)

OMPE command to generate an accounting trace report for the instance of the deadlock victim

```
//SYSIN    DD  *
DB2PM
* ********************************
*       GLOBAL PARMS
* ********************************
  GLOBAL
       TIMEZONE (+4)
       INCLUDE( DB2ID(D1B1) INSTANCE (CBBE4035CB4B))
* ********************************
*       ACCOUNTING REPORTS
* ********************************
  ACCOUNTING
    TRACE
    LAYOUT(LONG)
EXEC
```

Deadlock Scenario – Accounting Plan info of the Victim

TIMES/EVENTS	APPL(CL.1)	DB2 (CL.2)	IFI (CL.5)
ELAPSED TIME	1:01.84213	1:01.68192	N/P
NONNESTED	1.843288	1.683072	N/A
STORED PROC	57.966362	57.966362	N/A
UDF	0.000000	0.000000	N/A
TRIGGER	2.032485	2.032485	N/A
CP CPU TIME	33.310736	33.207112	N/P
AGENT	33.310736	33.207112	N/A
NONNESTED	0.258581	0.154957	N/P
STORED PRC	33.030787	33.030787	N/A
UDF	0.000000	0.000000	N/A
TRIGGER	0.021369	0.021369	N/A
PAR.TASKS	0.000000	0.000000	N/A
SECP CPU	0.000000	N/A	N/A
SE CPU TIME	0.007918	0.007875	N/A
NONNESTED	0.000079	0.000035	N/A
STORED PROC	0.007839	0.007839	N/A
UDF	0.000000	0.000000	N/A
TRIGGER	0.000000	0.000000	N/A
PAR.TASKS	0.000000	0.000000	N/A
SUSPEND TIME	0.000000	18.002590	N/A
AGENT	N/A	18.002590	N/A
PAR.TASKS	N/A	0.000000	N/A
STORED PROC	0.000000	N/A	N/A
UDF	0.000000	N/A	N/A
NOT ACCOUNT.	N/A	10.472181	N/A
...	...	...	...

CLASS 3 SUSPENSIONS	ELAPSED TIME	EVENTS
LOCK/LATCH(DB2+IRLM)	15.504227	207273
IRLM LOCK+LATCH	12.432272	60
DB2 LATCH	3.071955	207213
SYNCHRON. I/O	0.443729	365
DATABASE I/O	0.423457	325
LOG WRITE I/O	0.020272	40
OTHER READ I/O	0.455280	591
OTHER WRTE I/O	0.000865	2
SER.TASK SWTCH	0.617835	506
UPDATE COMMIT	0.567477	499
OPEN/CLOSE	0.000000	0
SYSLGRNG REC	0.000000	0
EXT/DEL/DEF	0.046282	1
OTHER SERVICE	0.004077	6
ARC.LOG(QUIES)	0.000000	0
LOG READ	0.000000	0
DRAIN LOCK	0.000000	0
CLAIM RELEASE	0.000000	0
PAGE LATCH	0.186376	18
NOTIFY MSGS	0.044179	4
GLOBAL CONTENTION	0.750097	2088
COMMIT PH1 WRITE I/O	0.000000	0
ASYNCH CF REQUESTS	0.000000	0

HIGHLIGHTS	
THREAD TYPE	ALLIED
TERM.CONDITION	NORMAL
INVOKE REASON	DEALLOC
PARALLELISM	NO
PCA RUP COUNT	0
RUP AUTONOM.TX	0
AUTONOMOUS TX	0
QUANTITY	0
COMMITS	498
ROLLBACK	3
SVPT REQUESTS	0
SVPT RELEASE	0
SVPT ROLLBACK	0
INCREM.BINDS	152
UPDATE/COMMIT	8.07
SYNCH I/O AVG.	0.001216
PROGRAMS	31
MAX CASCADE	2

Workload is driven by a (set of) batch job(s) committing frequently to get 'transactional behavior' but accounting info is for all of them so also for all IRLM L/L times and suspensions

Deadlock Scenario – Accounting Plan info of the Victim

```
 LOCATION: DB1B                    OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V5R2M0)                PAGE: 1-3
      GROUP: DB1BG                              ACCOUNTING TRACE - LONG                  REQUESTED FROM: NOT SPECIFIED
     MEMBER: D1B1                                                                                 TO: NOT SPECIFIED
  SUBSYSTEM: D1B1                                                                                              29:25.71
---- IDENTIFICATION ------------------------------------                                                      ------
ACCT TSTAMP: 07/31/13 15:29:25.71    PLANNAME: DSNREXX
BEGIN TIME : 07/31/13 15:28:23.87    PROD TYP: N/P
END TIME   : 07/31/13 15:29:25.71    PROD VER: N/P
REQUESTER  : DB1B                    CORRNAME: GLWRUN2
MAINPACK   : DSNREXX                 CORRNMBR: 'BLANK'        LUW INS: CBBE4035CB4B    ENDUSER : BART
PRIMAUTH   : BART                    CONNTYPE: DB2CALL        LUW SEQ:          502    TRANSACT: GLWRUN2
ORIGAUTH   : BART                    CONNECT : DB2CALL                                 WSNAME  : DB2CALL
```

Locking and data sharing locking section
Check #Lock vs #UNLOCK
Check suspensions

SQL DML	TOTAL	SQL DCL	TOTAL	SQL DDL	CREATE	DROP	ALTER
SELECT	8849	LOCK TABLE	0	TABLE	0	38	0
INSERT	3146	GRANT	0	CRT TTABLE	0	N/A	N/A
ROWS	4776	REVOKE	0	DCL TTABLE	38	N/A	N/A
UPDATE	890	SET SQLID	0	AUX TABLE	0	N/A	N/A
ROWS	856	SET H.VAR.	756	INDEX	0	0	0
MERGE	0	SET DEGREE	0	TABLESPACE	0	0	0
DELETE	6	SET RULES	0	DATABASE	0	0	0
ROWS	3	SET PATH	0	STOGROUP	0	0	0
		SET PREC.	0	SYNONYM	0	0	N/A
DESCRIBE	47	CONNECT 1	0	VIEW	0	0	0
DESC.TBL	0	CONNECT 2	0	ALIAS	0	0	N/A
PREPARE	70	SET CONNEC	0	PACKAGE	N/A	0	N/A
OPEN	157	RELEASE	0	PROCEDURE	0	0	0
FETCH	36675	CALL	3958	FUNCTION	0	0	0
ROWS	36577	ASSOC LOC.	0	TRIGGER	0	0	N/A
CLOSE	119	ALLOC CUR.	0	DIST TYPE	0	0	N/A
		HOLD LOC.	0	SEQUENCE	0	0	0
DML-ALL	49959	FREE LOC.	0	TRUST. CTX	0	0	0
		DCL-ALL	4714	ROLE	0	0	N/A

LOCKING	TOTAL
TIMEOUTS	0
DEADLOCKS	3
ESCAL.(SHAR)	0
ESCAL.(EXCL)	0
MAX PG/ROW LCK HELD	65
LOCK REQUEST	17498
UNLOCK REQST	1820
QUERY REQST	0
CHANGE REQST	1723
OTHER REQST	0
TOTAL SUSPENSIONS	68
LOCK SUSPENS	50
IRLM LATCH SUSPENS	10
OTHER SUSPENS	8

DATA SHARING	TOTAL
GLOB CONT(%)	0.02
FALS CONT(%)	0.02
P/L-LOCKS(%)	20
P-LOCK REQ	0
P-UNLOCK REQ	0
P-CHANGE REQ	0
LOCK - XES	3649
UNLOCK-XES	250
CHANGE-XES	592
SUSP - IRLM	0
SUSP - XES	0
CONV - XES	2088
FALSE CONT	1
INCOMP.LOCK	0
NOTIFY SENT	8

Acctg class 7

DPTDEL	
TYPE	PACKAGE
LOCATION	DB1B
COLLECTION ID	GLWSAMP
PROGRAM NAME	DPTDEL
CONSISTENCY TOKEN	1977A1EC16E3FDE8
ACTIVITY TYPE	NATIVE SQL PROC
ACTIVITY NAME	DPTDEL
SCHEMA NAME	GLWSAMP
SUCC AUTH CHECK	YES
NBR OF ALLOCATIONS	3
SQL STMT - AVERAGE	23.00
SQL STMT - TOTAL	23
NBR RLUP THREADS	1

DPTDEL	TIMES
ELAPSED TIME - CL7	2.615062
CP CPU TIME	0.023584
AGENT	0.023584
PAR.TASKS	0.000000
SE CPU TIME	0.000000
SUSPENSION-CL8	2.584659
AGENT	2.584659
PAR.TASKS	0.000000
NOT ACCOUNTED	0.006819

Acctg class 8

DPTDEL	TIME	EVENTS
LOCK/LATCH	2.459258	2
IRLM LOCK+LATCH	2.459258	2
DB2 LATCH	0.000000	0
SYNCHRONOUS I/O	0.061023	29
OTHER READ I/O	0.058544	43
OTHER WRITE I/O	0.000041	1
SERV.TASK SWITCH	0.002578	4
ARCH.LOG(QUIESCE)	0.000000	0
ARCHIVE LOG READ	0.000000	0
DRAIN LOCK	0.000000	0
CLAIM RELEASE	0.000000	0
PAGE LATCH	0.000000	0
NOTIFY MESSAGES	0.000000	0
GLOBAL CONTENTION	0.003214	17
TCP/IP LOB XML	0.000000	0
ACCELERATOR	0.000000	0
TOTAL CL8 SUSPENS.	2.584659	96

Acctg class 10

DPTDEL	TOTAL
TIMEOUTS	0
DEADLOCKS	2
ESCAL.(SHARED)	0
ESCAL.(EXCLUS)	0
MAX PG/ROW LOCKS HELD	65
LOCK REQUEST	872
UNLOCK REQUEST	115
QUERY REQUEST	0
CHANGE REQUEST	105
OTHER REQUEST	0
TOTAL SUSPENSIONS	2
LOCK SUSPENS	2
IRLM LATCH SUSPENS	0
OTHER SUSPENS	0

Looks like we had another deadlock for this package during this run

Simple Deadlock Scenario – Acctg record package info of victim

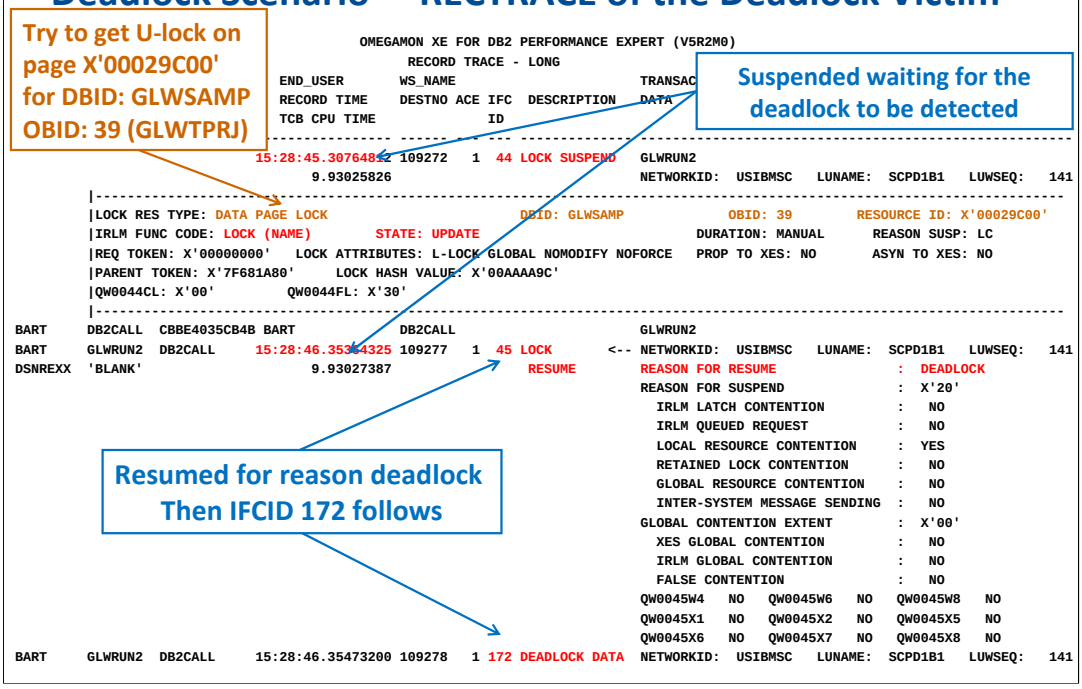
How Did We End Up with this Deadlock – Zoom in using detailed traces

- Looking at the
  - Deadlock record
  - SQL statements and
  - Programs involved
- May not allow to identify the root cause of the deadlock
- To resolve the problem
  - **Recreate the problem with a detailed trace active**
    (See before for the types of IFCIDs to collect)

How Did We End Up with this Deadlock – Zoom in using detailed traces -2

- To zoom in
- Run a detailed **RECTRACE of the threads involved**
  - **Start with the deadlock victim**
    - -911/-913 should be in the SQL trace record IFCID 58
    - IFCID 172 is for the victim, so it will also show in the victim's RECTRACE
    - For **each resource involved** in the deadlock
      - Start at the time the deadlock occurred
      - **Go back in time to see when the locks were requested**
      - Some locks will have been 'acquired', some locks we will be 'suspended for'
      - Go a bit more back in time to **find the SQL statement that resulted in the lock request** – The STMT_ID should match with info in IFCID 172

Deadlock Scenario – RECTRACE of the Deadlock Victim

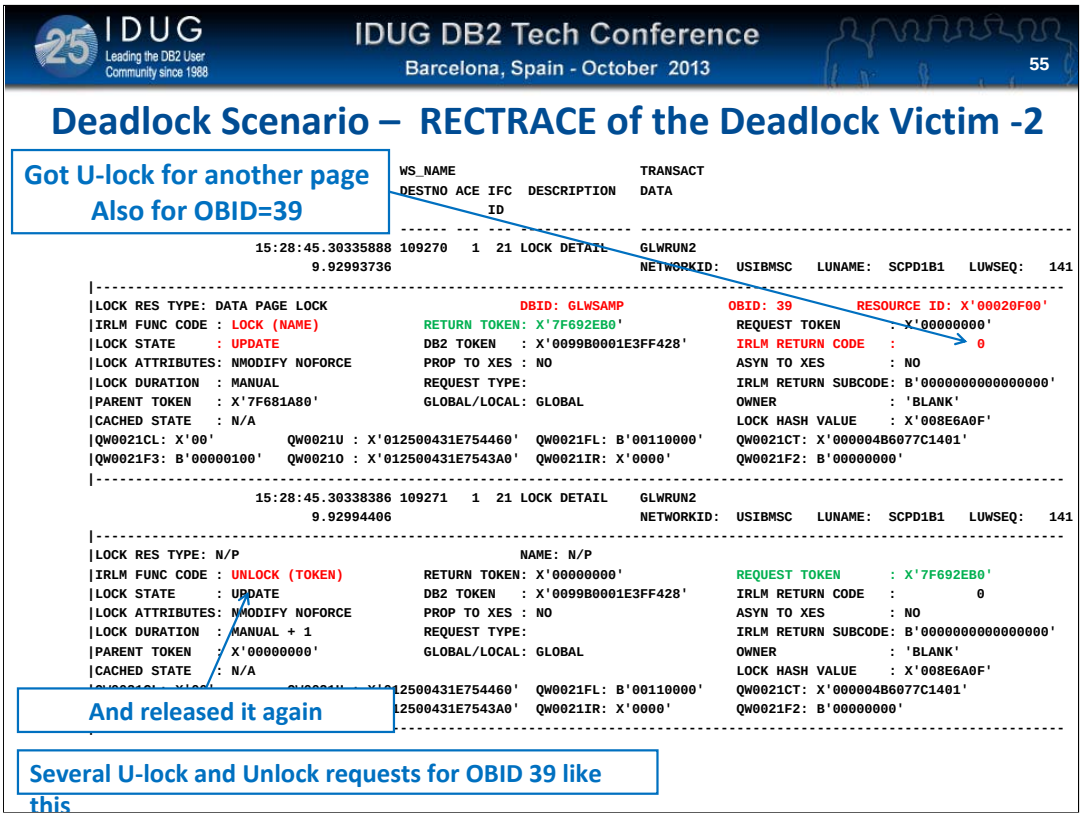



```
                    OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V5R2M0)
                          RECORD TRACE - LONG
END_USER       WS_NAME                        TRANSAC
RECORD TIME    DESTNO ACE IFC  DESCRIPTION    DATA
TCB CPU TIME              ID
-------------- ------ --- ---- -------------- ----------------------------------------------------------------------
                15:28:45.30764812 109272   1  44 LOCK SUSPEND   GLWRUN2
                     9.93025826                                  NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   141
        |-------------------------------------------------------------------------------------------------------------
        |LOCK RES TYPE: DATA PAGE LOCK                  DBID: GLWSAMP          OBID: 39        RESOURCE ID: X'00029C00'
        |IRLM FUNC CODE: LOCK (NAME)        STATE: UPDATE                 DURATION: MANUAL      REASON SUSP: LC
        |REQ TOKEN: X'00000000'   LOCK ATTRIBUTES: L-LOCK GLOBAL NOMODIFY NOFORCE   PROP TO XES: NO       ASYN TO XES: NO
        |PARENT TOKEN: X'7F681A80'     LOCK HASH VALUE: X'00AAAA9C'
        |QW0044CL: X'00'         QW0044FL: X'30'
        |-------------------------------------------------------------------------------------------------------------
BART    DB2CALL  CBBE4035CB4B BART            DB2CALL                     GLWRUN2
BART    GLWRUN2  DB2CALL      15:28:46.35364325 109277   1  45 LOCK      <-- NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   141
DSNREXX 'BLANK'                    9.93027387                RESUME       REASON FOR RESUME              :  DEADLOCK
                                                                          REASON FOR SUSPEND             :  X'20'
                                                                            IRLM LATCH CONTENTION        :   NO
                                                                            IRLM QUEUED REQUEST          :   NO
                                                                            LOCAL RESOURCE CONTENTION    :  YES
                                                                            RETAINED LOCK CONTENTION     :   NO
                                                                            GLOBAL RESOURCE CONTENTION   :   NO
                                                                            INTER-SYSTEM MESSAGE SENDING :   NO
                                                                          GLOBAL CONTENTION EXTENT       :  X'00'
                                                                            XES GLOBAL CONTENTION        :   NO
                                                                            IRLM GLOBAL CONTENTION       :   NO
                                                                            FALSE CONTENTION             :   NO
                                                                          QW0045W4   NO   QW0045W6   NO   QW0045W8   NO
                                                                          QW0045X1   NO   QW0045X2   NO   QW0045X5   NO
                                                                          QW0045X6   NO   QW0045X7   NO   QW0045X8   NO
BART    GLWRUN2  DB2CALL      15:28:46.35473200 109278   1 172 DEADLOCK DATA NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   141
```

Start with the deadlock time and work your way back in time to find the resource that the thread was suspended for (and resulted in the deadlock)

You can use the following OMPE command to created the RECTRACE output DB2PM

```
* ********************************
*      GLOBAL PARMS
* ********************************
 GLOBAL
   FROM(, 15:28:45.25)
   TO(, 15:28:46.36)
   TIMEZONE (+4)
   INCLUDE( DB2ID(D1B1) INSTANCE (CBBE4035CB4B))
* ********************************
*      RECORD TRACES
* ********************************
 RECTRACE
    TRACE
     LEVEL(LONG)
EXEC
```



Going back in time, we see that the job requested a similar U-lock but on a different page of OBID=39

That request was successul (irlm return code = 0)

Next we remove the lock again (via UNLOCK).

Note that we use the request token for the unlock, not the resource name like we did for the lock request.

In the trace there are more similar lock/unlock requests

(going through the GLWTPRJ table for matching department numbers, to delete (to enforce the RI constraint)

Deadlock Scenario – RECTRACE of the Deadlock Victim -3

```
LOCATION: DB1B                          OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V5R2M0)            PAGE: 1-19
      GROUP: DB1BG                              RECORD TRACE - LONG                          REQUESTED FROM: ALL      15:28:45.25
     MEMBER: D1B1                                                                                      TO: DATES    15:28:46.36
  SUBSYSTEM: D1B1                                                                                ACTUAL FROM: 07/31/13 15:28:45.25
DB2 VERSION: V11                                                                                   PAGE DATE: 07/31/13
PRIMAUTH CONNECT   INSTANCE      END_USER       WS_NAME                      TRANSACT
ORIGAUTH CORRNAME  CONNTYPE      RECORD TIME    DESTNO ACE IFC  DESCRIPTION  DATA
PLANNAME CORRNMBR                TCB CPU TIME             ID
-------- -------- ----------- ---------------- ------ --- --- -------------- --------------------------------------------------
BART     DB2CALL  CBBE4035CB4B BART            DB2CALL                      GLWRUN2
BART     GLWRUN2  DB2CALL     15:28:45.25323556 108925   1  61 DELETE     --> NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   141
DSNREXX  'BLANK'                    9.92034210                    START        LOCATION NAME        : DB1B
                                                                               PKG COLLECTION ID    : GLWSAMP
                                                                               PROGRAM NAME         : DPTDEL
                                                                               CONSISTENCY TOKEN    : X'1977A1EC16E3FDE8'
                                                                               STATEMENT NUMBER     :          1
                                                                               STATEMENT TYPE       : DELETE TYPE- NON CURSOR
                                                                               CURSOR NAME          : N/P
                                                                               QUERY COMMAND ID     : N/P
                                                                               QUERY INSTANCE ID    : N/P
                                                                               ISOLATION            : CS
                                                                               REOPTIMIZATION       : NO
                         15:28:45.25327713 108926   1 211 CLAIM DATA          GLWRUN2
                               9.92036821                                      NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   141
        |-------------------------------------------------------------------------------------------------------------------------------
        |DBID:   GLWSAMP    PSID:  GLWSDPT       PARTITION NO.:    0    CLAIM REQUEST TYPE: ACQUIRE   CLAIM CLASS: WRITE
        |CLAIM DURATION: HELD UNTIL COMMIT   CLAIM RESULT: SUCCESSFUL
        |-------------------------------------------------------------------------------------------------------------------------------
```

This is the SQL stmt that triggered the U-lock(s)
Matches with what we found using the STMT_ID

Starts with GLWSDPT

OBID=3 is GLWTDPT

Deadlock Scenario – RECTRACE of the Deadlock Victim -4

Got a U-lock here for the 'other' resource of the deadlock but we were holding an X-lock at time of deadlock ??

```
PRIMAUTH CONNECT
ORIGAUTH CORRNAME
PLANNAME CORRNMBR
-------- --------

                          15:28:45.25404292 108999   1  21 LOCK DETAIL    GLWRUN2
                                9.92073739                                NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   141
        |-------------------------------------------------------------------------------------------------------------------------
        |LOCK RES TYPE: DATA PAGE LOCK                        DBID: GLWSAMP             OBID: GLWSEMP   RESOURCE ID: X'003000FC00'
        |IRLM FUNC CODE : LOCK (NAME)          RETURN TOKEN: X'7F692E50'               REQUEST TOKEN      : X'00000000'
        |LOCK STATE     : UPDATE               DB2 TOKEN   : X'0099B0001E3FF428'       IRLM RETURN CODE   :          0
        |LOCK ATTRIBUTES: NMODIFY NOFORCE      PROP TO XES : NO                        ASYN TO XES        : NO
        |LOCK DURATION  : MANUAL               REQUEST TYPE:                           IRLM RETURN SUBCODE: B'0000000000000000'
        |PARENT TOKEN   : X'7F696F10'          GLOBAL/LOCAL: GLOBAL                    OWNER              : 'BLANK'
        |CACHED STATE   : N/A                                                          LOCK HASH VALUE    : X'0035E8FC'
        |QW0021CL: X'00'         QW0021U : X'012500431E754460'  QW0021FL: B'10110000'   QW0021CT: X'000004B6077C1401'
        |QW0021F3: B'00000100'   QW0021O : X'012500431E7543A0'  QW0021IR: X'0000'       QW0021F2: B'00000000'
        |-------------------------------------------------------------------------------------------------------------------------
… … … … … … … … … … …
… … … … … … … … … … …
                          15:28:45.25407335 109002   1  21 LOCK DETAIL    GLWRUN2
                                9.92076426                                NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   141
        |-------------------------------------------------------------------------------------------------------------------------
        |LOCK RES TYPE: N/P                                   NAME: N/P
        |IRLM FUNC CODE : CHANGE (TOKEN)       RETURN TOKEN: X'7F692E50'               REQUEST TOKEN      : X'7F692E50'
        |LOCK STATE     : EXCLUSIVE            DB2 TOKEN   : X'0099B0001E3FF428'       IRLM RETURN CODE   :          0
        |LOCK ATTRIBUTES: MODIFY  NOFORCE      PROP TO XES : NO                        ASYN TO XES        : NO
        |LOCK DURATION  : COMMIT               REQUEST TYPE:                           IRLM RETURN SUBCODE: B'0000000000000000'
        |PARENT TOKEN   : X'7F696F10'          GLOBAL/LOCAL: GLOBAL                    OWNER              : 'BLANK'
        |CACHED STATE   : N/A                                                          LOCK HASH VALUE    : X'0035E8FC'
                                                 754460'  QW0021FL: B'00110010'   QW0021CT: X'000004B6077C1401'
                                                 543A0'   QW0021IR: X'0000'       QW0021F2: B'00010000'
                                                                                  -------------------------------------------------
```

Changed to X (successful) by using the lock token not the resouce name

So when did we acquire the other lock that we were 'holding" when the deadlock was detected.

X-lock on GLSWSEMP page x'003000FC'

Go back to the deadlock time and seach backward for the resource name.

Using the resource name there is only a U-lock requested

So where it the X-lock ??

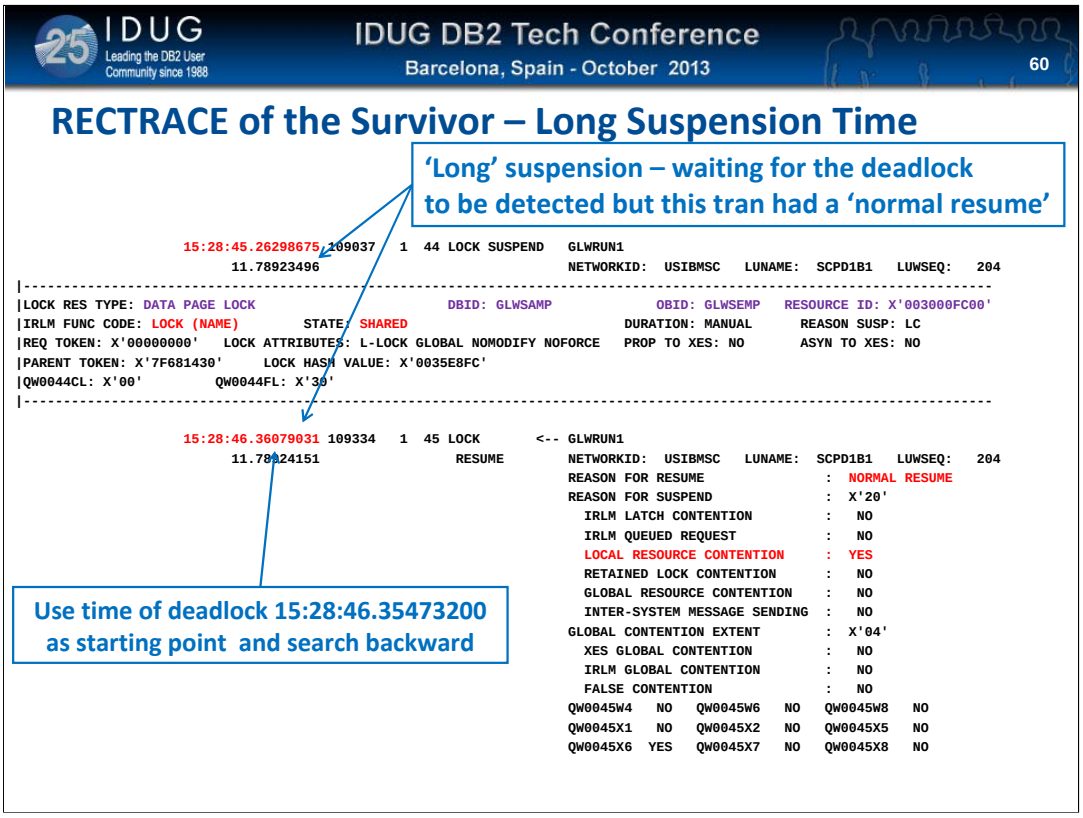
Use the lock token to search (instead of the lock name – db.ts.page) and find eht change to exclusive.

Simple Deadlock Scenario – The victim's story

- **Scenario for the victim** that we constructed:
  - 15:28:45.25323556 - DELETE FROM GLWTDPT WHERE DEPT_NO = :hv
    - Delete removes dept and needs to enforce the 2 RI constraints
  - 15:28:45.25404292 – U-lock DBID: GLWSAMP OBID: GLWSEMP RESOURCE ID: X'003000FC00' (RI#1 for emp)
  - 15:28:45.25407335 - Upgrade to X-lock via request token
    - Tran does some more checks
    - Then starts checking the other RI relationship
  - 15:28:45.3076481 - Try to lock DATA PAGE DBID: GLWSAMP OBID: 39 RESOURCE ID: X'00029C00' (RI#2 GLWTPRJ) and get suspended as the lock is not available
  - 15:28:46.35364325 – Lock resume with reason deadlock
    - This tran is selected as the victim of the deadlock
  - It will resume (with reason=deadlock) and will issue a rollback (not shown)

Simple Deadlock Scenario – How Did We End Up with this Deadlock – Zoom in using detailed trace - 2

- Run a detailed RECTRACE of the threads involved – cont.
  - Do the same for **other transactions involved in the deadlock**
    - There will be no -911/-913 or IFCID 172 for those (survived the deadlock)
    - Use timestamp of IFCID 172 (deadlock) to work your way back
    - Just after the deadlock victim was selected, tran surviving the deadlock should be resumed (IFCID 45 and IFCID 21)
      - With > 2 tran involved it is possible that some tran remain suspended after the deadlock occurred , if there are other waiters 'ahead' of them
    - **For each resource involved in the deadlock analyze when/if the lock was obtained and the SQL statement that triggered it**
      - Same analysis as the deadlock victim (see before)



As this is the surving tran, it will not have the deadlock record itself.

However, the deadlock time is a good starting point for our investigations

Note the long suspension time – from `15:28:45.26298675 till 15:28:46.36079031`

`Remember that the deadlock detection cycle is 1 second on this system.`

RECTRACE of the Survivor – Find SQL

```
15:28:45.18613343 108528   1  60 SELECT     --> GLWRUN1
      11.77302817                 START          NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   204
                                                 LOCATION NAME        : DB1B
                                                 PKG COLLECTION ID    : GLWSAMP
                                                 PROGRAM NAME         : EMPSEL
                                                 CONSISTENCY TOKEN    : X'1977A1EC125C0A1C'
                                                 STATEMENT NUMBER     :          1
                                                 STATEMENT TYPE       : X'00'
                                                 QUERY COMMAND ID     : N/P
                                                 QUERY INSTANCE ID    : N/P
                                                 ISOLATION            : CS
                                                 REOPTIMIZATION       : NO
```

Go back in time to find the start of the SQL Statement

And continue to go back in time until you find the SQL stmt that resulted in the lock being acquired. It is the start of the SQL statement.

RECTRACE of the Survivor – Find End SQL

```
BART     DB2CALL  CBBE4031CD7F BART            DB2CALL                  GLWRUN1
BART     GLWRUN1  DB2CALL     15:28:46.37755738 109841   1  58 END SQL    <-- NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   204
DSNREXX  'BLANK'                   11.80248161
         |-------------------------------------
         |LOCATION NAME         : DB1B
         |PKG COLLECTION ID     : GLWSAMP
         |PROGRAM NAME          : EMPSEL
         |CONSISTENCY TOKEN     : X'1977A1EC125C0A
         |STATEMENT NUMBER      :          1
         |
         |SQLCAID: SQLCA         SQLCABC          136                    SQLERRP : DSN            SQLEXT  :   00000
         |SQLERRD1          0   SQLERRD2           0   SQLERRD3           0   SQLWARN0:     SQLWARN1:     SQLWARN2:    SQLWARN3:
                           -1   SQLERRD5           0   SQLERRD6           0   SQLWARN4:     SQLWARN5:     SQLWARN6:    SQLWARN7:
         |.....................................................................................................................
         |DATA TYPE        INDX ROW PROC          83 ROW EXAM         82 STG1-QUAL          1 STG2-QUAL          1 ROW INSRT         0
         |ROW UPDTE           0 ROW DELET          0 PAGES             83 RI SCAN            0 RI DELET           0
         |LOB SCAN            0 LOB UPDTE          0
         |DATA TYPE       SEQD ROW PROC       24669 ROW EXAM      24669 STG1-QUAL          0 STG2-QUAL          0 ROW INSRT         0
         |ROW UPDTE           0 ROW DELET          0 PAGES         24293 RI SCAN            0 RI DELET           0
         |LOB SCAN            0 LOB UPDTE          0
         |.....................................................................................................................
         |TYPE OF STATEMENT       : STATIC                        STATEMENT IDENTIFIER    :              6982
         |NBR OF SYNC READS       :                    0          NBR OF GETPAGES         :             24376
         |NBR OF ROWS EXAMINED    :                24669          NBR OF ROWS PROCESSED   :                 0
         |NBR OF SORTS            :                    0          NBR OF INDEX SCANS      :                 1
         |NBR OF TABLESPACE SCANS :                    0          NBR OF BUFFER WRITES    :                 0
         |NBR OF PAR. GRPS CREATED:                    0
         |ACCUMULATED TIME VALUES:
         |IN-DB2 ELAPSED          :         1.191416               IN-DB2 CPU              :         0.029447
         |WAIT FOR SYNC I/O       :              N/P               WAIT FOR LOCK/LATCH     :         1.097804
         ......
```

'Long' suspension and stmt_id (IFCID 400 info)
SELECT MIN(EMP_NO) INTO :H:H FROM
GLWTEMP WHERE EMP_NO >= :H:H AND
MANAGER = :H:H

That's a lot !

After we were resumed the statement completed successfully (SQLCODE=0) .

The END SQL trace record (IFCID 58) has a lot of interesting information.

IFCID 400 is new in DB2 10 and contains similar info to the DSC info for dynamic SQL, but IFCID 400 is for static SQL

When IFCID 400 is active, IFCID 58 (end SQL) also contains additional information (as shown on the foil).

RECTRACE of the Survivor – STMT_ID d'7302'- x'1C86'

```
15:28:44.98365122 106831   1  21 LOCK DETAIL    GLWRUN1
                          11.70963003                          NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   204
|-------------------------------------------------------------------------------------------------------------------
|LOCK RES TYPE: DATA PAGE LOCK                    DBID: GLWSAMP            OBID: 39        RESOURCE ID: X'00029C00'
|IRLM FUNC CODE : LOCK (NAME)          RETURN TOKEN: X'FFFFFFFF'           REQUEST TOKEN       : X'00000000'
|LOCK STATE     : SHARED               DB2 TOKEN   : X'0099B0001E3FF428'   IRLM RETURN CODE   :          4
|LOCK ATTRIBUTES: NMODIFY NOFORCE      PROP TO XES : NO                    ASYN TO XES         : NO
|LOCK DURATION  : COMMIT               REQUEST TYPE:                       IRLM RETURN SUBCODE: B'0001000000000000'
|PARENT TOKEN   : X'7F69A2E0'          GLOBAL/LOCAL: GLOBAL                OWNER               : 'BLANK'
|CACHED STATE   : N/A                                                      LOCK HASH VALUE     : X'00AAAA9C'
|QW0021CL: X'00'         QW0021U : X'012500401E754180'  QW0021FL: B'11110000'   QW0021CT: X'000004B6077CE501'
|QW0021F3: B'00000001'   QW0021O : X'012500401E7540C0'  QW0021IR: X'0000'       QW0021F2: B'00000000'
|-------------------------------------------------------------------------------------------------------------------
```

DBID: GLWSAMP OBID: 39 RESOURCE ID: X'00029C00' is our resource but the request is for an S-lock (and we have X) and also irlm RC=4

LOCK request

The following table shows the return
LOCK requests:

Return code	Reason code (byte 1)	Reason code (byte 2)	Descriptio
04 (X'04')	X'80'		A restart lock successfully reacquired a retained lock.
	X'40'		A modify lock was granted, but no Record List Entry (RLE) was requested in the CF.
	X'20'		Other lock holders exist.
	X'10'		The lock is already held.
	X'08'		This lock was involved in a deadlock, but was not selected as a victim.
	X'04'		Another work unit holds the lock private.
		X'04'	At least one other work unit holds the lock with the repeatable read (RR) attribute.
		X'02'	The work unit that submitted this lock request holds the lock with the repeatable read (RR) attribute.

Find the other resource that was involved in the deadlock and how the lock was acquired by the deadlock survivor.

Searching backward from the resume time (normal resume after the deadlock was broken) , search for the other resource name that was involved in the deadlock.

Check the "IRLM Messages and Codes for IMS and DB2 for z/OS" - GC19-2666

- Chapter 2. IRLM return and reason codes for "the LOCK request" RC and RSN

RECTRACE of the Survivor – Stmt_ID d'7302'- x'1C86' -2

Read bottom-up

INSERT INTO GLWTPRJ

```
DB2 VERSION: V11
PRIMAUTH CONNECT   INSTANCE      END_USER       WS_NAME                    TRANSACT
ORIGAUTH CORRNAME  CONNTYPE      RECORD TIME    DESTNO ACE IFC  DESCRIPTION DATA
PLANNAME CORRNMBR                TCB CPU TIME             ID
-------- -------- ----------- ---------------- ------ --- --- ------------ ------------------------------------------------------------
BART     DB2CALL  CBBE4031CD7F BART             DB2CALL                    GLWRUN1
BART     GLWRUN1  DB2CALL     15:28:44.96896189 106738   1  61 INSERT     --> NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   204
DSNREXX  'BLANK'                   11.70623593                  START        LOCATION NAME        : DB1B
                                                                             PKG COLLECTION ID    : GLWSAMP
                                                                             PROGRAM NAME         : PRJADD
                                                                             CONSISTENCY TOKEN    : X'1977A1ED017C6108'
                                                                             STATEMENT NUMBER     :          1
                                                                             STATEMENT TYPE       : INSERT TYPE
                                                                             CURSOR NAME          : N/P
                                                                             QUERY COMMAND ID     : N/P
                                                                             QUERY INSTANCE ID    : N/P
                                                                             ISOLATION            : CS
                                                                             REOPTIMIZATION       : NO
...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...  ...
BART     DB2CALL  CBBE4031CD7F BART             DB2CALL                    GLWRUN1
BART     GLWRUN1  DB2CALL     15:28:44.97010577 106749   1  21 LOCK DETAIL  NETWORKID:  USIBMSC   LUNAME:  SCPD1B1   LUWSEQ:   204
DSNREXX  'BLANK'                   11.70628973
         |-------------------------------------------------------------------------------------------------------------------------
         |LOCK RES TYPE: DATA PAGE LOCK                    DBID: GLWSAMP          OBID: 39       RESOURCE ID: X'00029C00'
         |IRLM FUNC CODE : LOCK (NAME)          RETURN TOKEN: X'7F693AC0'          REQUEST TOKEN      : X'00000000'
         |LOCK STATE     : EXCLUSIVE            DB2 TOKEN   : X'0099B0001E3FF428'  IRLM RETURN CODE   :          0
         |LOCK ATTRIBUTES: MODIFY  NOFORCE      PROP TO XES : NO                   ASYN TO XES        : NO
         |LOCK DURATION  : COMMIT              REQUEST TYPE:                      IRLM RETURN SUBCODE: B'0000000000000000'
         |PARENT TOKEN   : X'7F69A2E0'         GLOBAL/LOCAL: GLOBAL               OWNER              : 'BLANK'
         |CACHED STATE   : N/A                                                    LOCK HASH VALUE    : X'00AAAA9C'
         |QW0021CL: X'00'         QW0021U : X'012500401E754180'  QW0021FL: B'10110010'   QW0021CT: X'000004B6077CE501'
         |QW0021F3: B'00000100'   QW0021O : X'012500401E7540C0'  QW0021IR: X'0000'       QW0021F2: B'00000000'
         |-------------------------------------------------------------------------------------------------------------------------
```

Go back in time some more , and there we find the X-lock (IRLM RC=0 this time (as it should))

Go back some more and find the SQL (ISRT) that triggered the X-lock to be acquired.

Simple Deadlock Scenario – The Survivor's Story

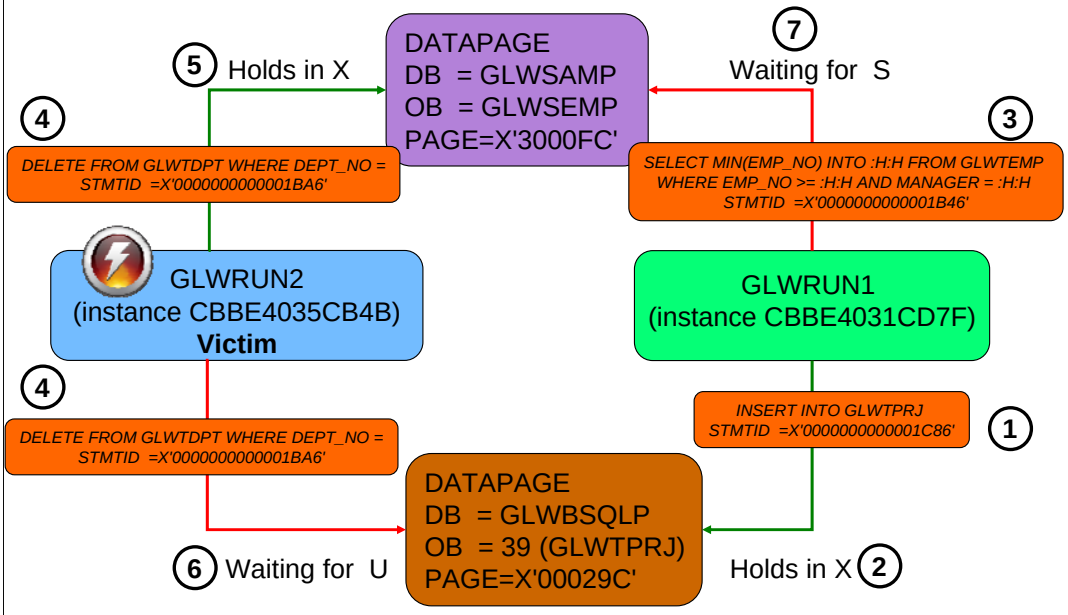
- Scenario for the survivor that we constructed:
  - 15:28:44.96896189 – INSERT INTO GLWTPRJ
    - Inserts new project
  - 15:28:44.97010577 - X lock PAGE DBID: GLWSAMP OBID: 39 (GLWTPRJ)
    RESOURCE ID: X'00029C00'
  - 15:28:44.97758161 - INSERT ends (not committed yet)
  - Inserting a project requires other things to be done than just adding a row in the project table - so more SQL has to be executed – one of them is this one
  - 15:28:45.18613343 - SELECT starts
    SELECT MIN(EMP_NO) INTO :H:H FROM GLWTEMP
    WHERE EMP_NO >= :H:H AND MANAGER = :H:H
  - 15:28:45.26298675 – Suspends for S-lock on DBID:GLWSAMP OBID: GLWSEMP
    RESOURCE ID: X'003000FC00'
    - We are in a deadlock situation now , and start to wait for IRLM and DB2 to detect and decide who to 'kill'
  - 15:28:46.36079031 – Tran is resumed (we were not picked as deadlock victim)

Simple Deadlock Scenario – The complete picture

Survivor
Victim

1. 15:28:44.96896189 - INSERT INTO GLWTPRJ
2. 15:28:44.97010577 - X lock PAGE DBID: GLWSAMP OBID: 39 (GLWTPRJ)
   RESOURCE ID: X'00029C00'
- 15:28:44.97758161 - INSERT ends (not committed yet)
3. 15:28:45.18613343 - SELECT starts
   SELECT MIN(EMP_NO) INTO :H:H FROM GLWTEMP
   WHERE EMP_NO >= :H:H AND MANAGER = :H:H
4. 15:28:45.25323556 - DELETE FROM GLWTDPT WHERE DEPT_NO = :hv
- 15:28:45.25404292 - U-lock DBID: GLWSAMP OBID: GLWSEMP
  RESOURCE ID: X'003000FC00'
5. 15:28:45.25407335 - Upgrade to X-lock via request token
6. 15:28:45.30764810 - Try to lock DATA PAGE DBID: GLWSAMP OBID: 39
   RESOURCE ID: X'00029C00' - get suspended
7. 15:28:45.26298675 - Suspends for S-lock on DBID:GLWSAMP OBID: GLWSEMP
   RESOURCE ID: X'003000FC00
- 15:28:46.35364325 - Lock resume wit Reason deadlock
- 15:28:46.36079031 - Tran is resumed (not picked as the deadlock victim)

Simple Deadlock Scenario – The complete picture

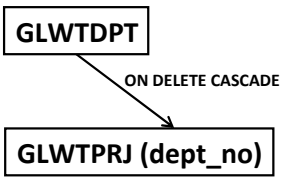


Simple Deadlock Scenario – Solution

- SELECT MIN(EMP_NO) INTO :H:H FROM GLWTEMP WHERE EMP_NO >= :H:H AND MANAGER = :H:H
  - Expensive SQL statement (NBR OF ROWS EXAMINED : 24669)
  - Inefficient access path ?
    - Create additional index to speed up the stmt
    - CREATE UNIQUE INDEX GLWSAMP.GLWXEMP4
      ON GLWSAMP.GLWTEMP
      (EMP_NO DESC,
      MANAGER ASC)
- This should take care of the deadlock !
- Or not ?
- Please place your bets now !

Simple Deadlock Scenario – Solution - 2

- It did NOT fix the problem - So where is the problem ?
- Remember the RI checks ?
  - There were many U-lock and unlock requests when doing the RI checks against GLWTPRJ – OBID=39 to enforce FK GLWFPRJ1
  - ALTER TABLE GLWSAMP.GLWTPRJ
    FOREIGN KEY GLWFPRJ1 (DEPT_NO)
    REFERENCES GLWSAMP.GLWTDPT (DEPT_NO)
    ON DELETE CASCADE ;



  - Does this FK have an index ?
    - No ! Oops !
    - CREATE INDEX GLWSAMP.GLWXPRJ2
      ON GLWSAMP.GLWTPRJ
      (DEPT_NO ASC)
- Will this do the Trick ?
  - Yes it did – After this, the workload showed no -913 anymore

Yes, addin the IX to the FK did it – no more deadlocks afterwards.

Simple Deadlock Scenario – Solution

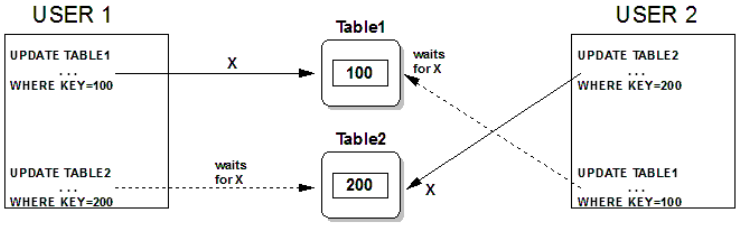
- Once you understand what triggers the deadlock the solution is often straight forward – like this case
- Other times application changes are required to avoid deadlocks

Deadlock Avoidance Guidelines

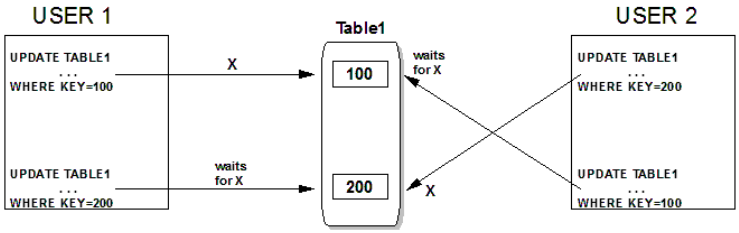
- As a general rule, consider the following procedures to avoid deadlocks
- **Reduce the lock duration**
  - Commit as frequently as possible
  - Use 'lowest' possible isolation level
- **Use ordered updates on different kinds of resources** that cannot detect each other's deadlock situations, e.g.. between DB2 and IMS
- **Use ordered updates within the same kind of resources** (within a DB2 subsystem)
  - Use ordered access to the different tables involved
  - Use ordered access to the rows within the tables

Deadlock example caused by 'unordered' access

- Unordered table access



- Unordered row access



Note: The access path is assumed to be matching index access

Deadlock Avoidance Guidelines -2

- Care should be taken in multi-table SQL statements in concurrent programs where the join order can change between binds
- Consider row-level locking if different applications are accessing different rows
  - Be aware of page P-lock in data sharing when using RLL
- Code cursors unambiguously, specifically stating FOR FETCH ONLY when appropriate
- Code for good lock avoidance
- More/faster resources (like CPUs) typically will not solve locking problems
- **Locking should NOT be an afterthought –**
  It should be part of the application design

Summary

- Don’t be afraid of locking problems
- Don‘t be afraid of DB2 detailed traces
- Good bedtime reading:
  - DB2 9 for z/OS: Resource Serialization and Concurrency Control (SG24-4725-1)

Thanks for attending

Bart Steegmans
IBM
bart_steegmans@be.ibm.com

Session : A13

Analyzing DB2 for z/OS Resource Serialization and Concurrency Problems

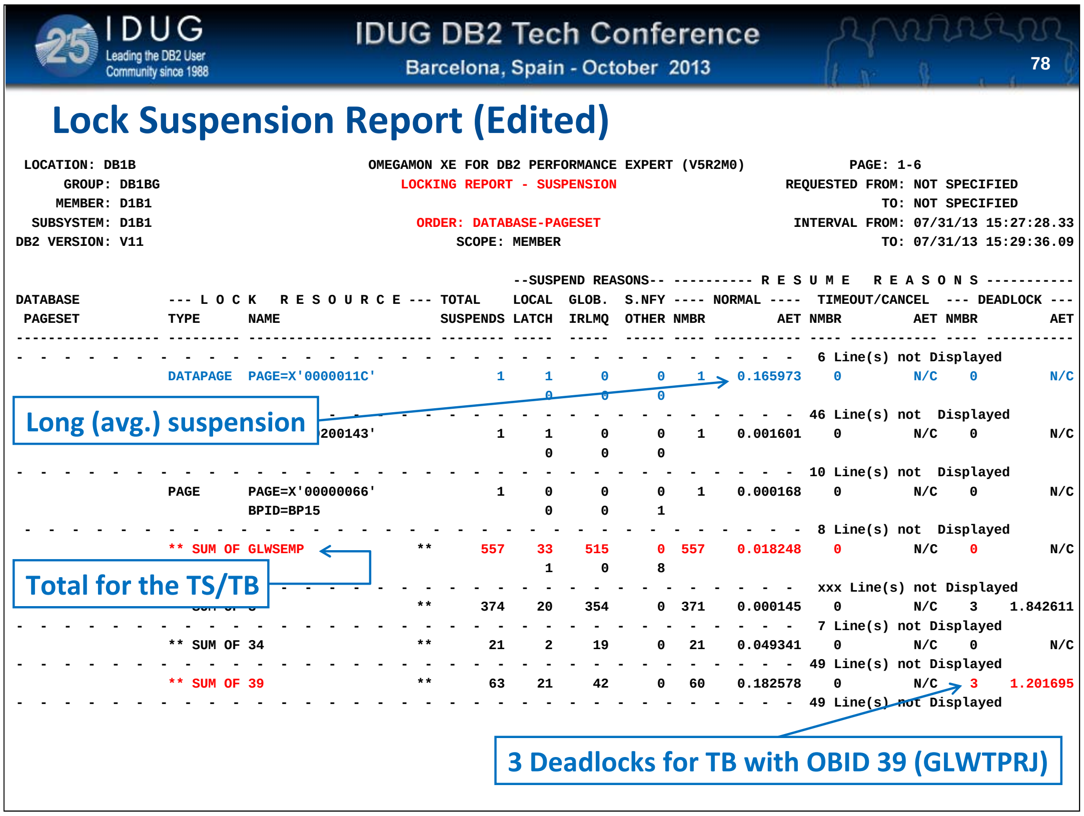
INTERNATIONAL DB2 USERS GROUP
Independent • Not-for-Profit • User Run
Founded 1988

Additional Slides

- In case there is time left ☺
  - That would be the first time !
- Analyzing Long Suspension Times
  - Accounting information
  - Lock suspension traces and reports
  - Detailed lock reports

Analyzing Long Suspension Times

- Most often triggered by a complaint – slow response time
- Look at the DB2 accounting data
  - High suspension time – avg. and/or total
    - LOCK/LATCH wait time
    - Global lock suspension
    - [Page latch suspension]
- Get detailed trace
  - 'Lock suspension' trace is enough to find out the resources involved
- Create lock suspension report
  - Find resources with high avg. Suspension time - or many suspensions
- Create lock suspension trace for those resources
  - Find long suspensions
- Get a more detailed SQL and locking trace to zoom in more




```
 LOCATION: DB1B                             OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V5R2M0)             PAGE: 1-6
      GROUP: DB1BG                              LOCKING REPORT - SUSPENSION                     REQUESTED FROM: NOT SPECIFIED
     MEMBER: D1B1                                                                                           TO: NOT SPECIFIED
  SUBSYSTEM: D1B1                                  ORDER: DATABASE-PAGESET                       INTERVAL FROM: 07/31/13 15:27:28.33
DB2 VERSION: V11                                        SCOPE: MEMBER                                       TO: 07/31/13 15:29:36.09

                                                                --SUSPEND REASONS-- ---------- R E S U M E   R E A S O N S -----------
DATABASE           --- L O C K   R E S O U R C E --- TOTAL    LOCAL  GLOB.  S.NFY ---- NORMAL ----  TIMEOUT/CANCEL  --- DEADLOCK ---
 PAGESET           TYPE      NAME                    SUSPENDS LATCH  IRLMQ  OTHER NMBR        AET NMBR         AET NMBR         AET
------------------ --------- ----------------------- -------- -----  -----  ----- ---- ----------- ---- ----------- ---- -----------
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -   6 Line(s) not Displayed
                   DATAPAGE  PAGE=X'0000011C'               1     1      0      0    1    0.165973    0        N/C    0         N/C
                                                                  0      0      0
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  46 Line(s) not  Displayed
                             200143'                        1     1      0      0    1    0.001601    0        N/C    0         N/C
                                                                  0      0      0
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  10 Line(s) not  Displayed
                   PAGE      PAGE=X'00000066'               1     0      0      0    1    0.000168    0        N/C    0         N/C
                             BPID=BP15                            0      0      1
 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  8 Line(s) not  Displayed
                   ** SUM OF GLWSEMP              **      557    33    515      0  557    0.018248    0        N/C    0         N/C
                                                                  1      0      8
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  xxx Line(s) not Displayed
                                                  **      374    20    354      0  371    0.000145    0        N/C    3    1.842611
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -   7 Line(s) not Displayed
                   ** SUM OF 34                   **       21     2     19      0   21    0.049341    0        N/C    0         N/C
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  49 Line(s) not Displayed
                   ** SUM OF 39                   **       63    21     42      0   60    0.182578    0        N/C    3    1.201695
-  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  49 Line(s) not Displayed
```

(The report has been edited to fit a bit better on the foil.)

This reports shows IRLM latch contentions, IRLM local and global lock contention and DB2 page latch contention (BM).

It does not report on DB2 internal latch contentions.

The "** SUM OF" gives the totals per TS/TB. This is usually a good way to find the objects with most/longest suspensions.

For example GLWSEMP had 557 suspensions during the trace time, 33 local lock suspension and 515 global lock suspensions.

The avg ET per suspension is 0.018 sec.

Moving up a bit in the output, there are all the individual pages/rows/.. that suspensions occurred for.

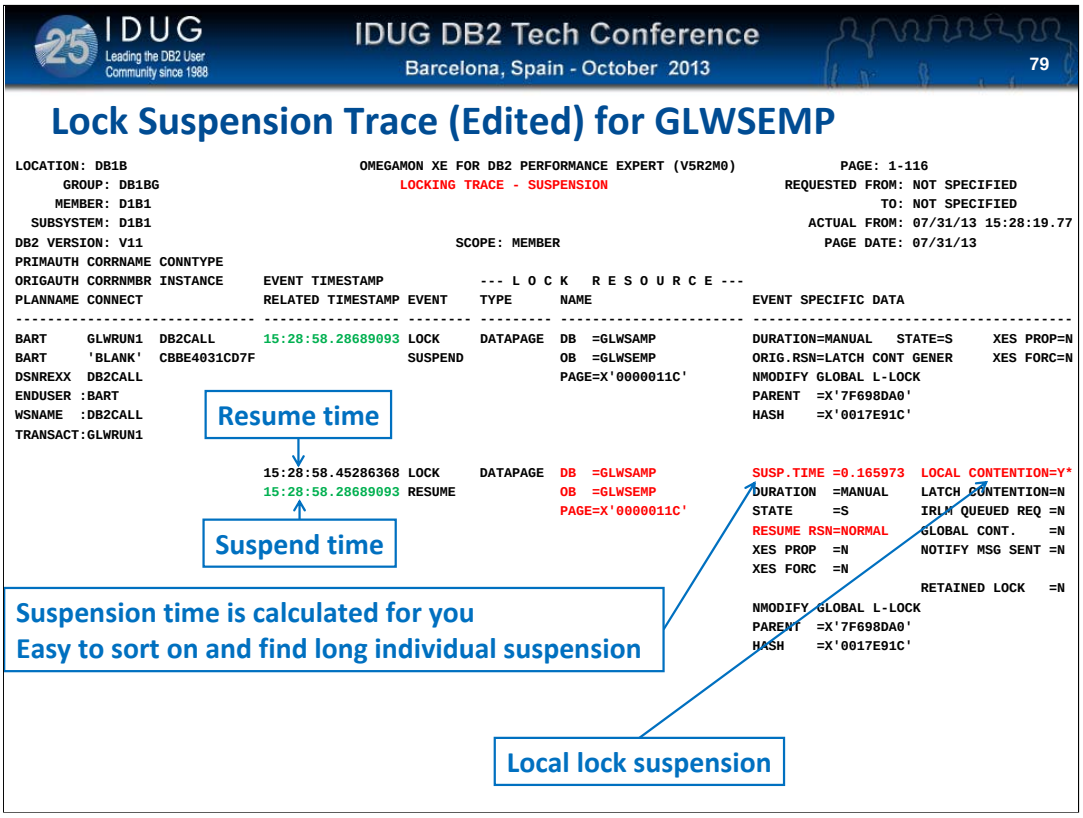
For example page x'0000011C' had one local lock suspension that lasted 0.165 seconds.

(Long is always relative of course, but given that the transactions are normally only a few msec, 165 msec is a long time.)

Note also the 3 deadlocks for OBID 39 in the lock suspension report – one of which we analyzed in detail earlier in the presentation.

You can use the following //SYSIN DD * to create an OMPE lock suspension report:

```
DB2PM
* *********************************
*     GLOBAL PARMS
* *********************************
 GLOBAL
     TIMEZONE (+4)
     INCLUDE( DB2ID(D1B1))
* *********************************
*     LOCKING REPORT
* *********************************
 LOCKING
        REPORT
        LEVEL(SUSPENSION)
        ORDER(DATABASE-PAGESET)
EXEC
```



The report has been edited to fit a bit better on the foil.

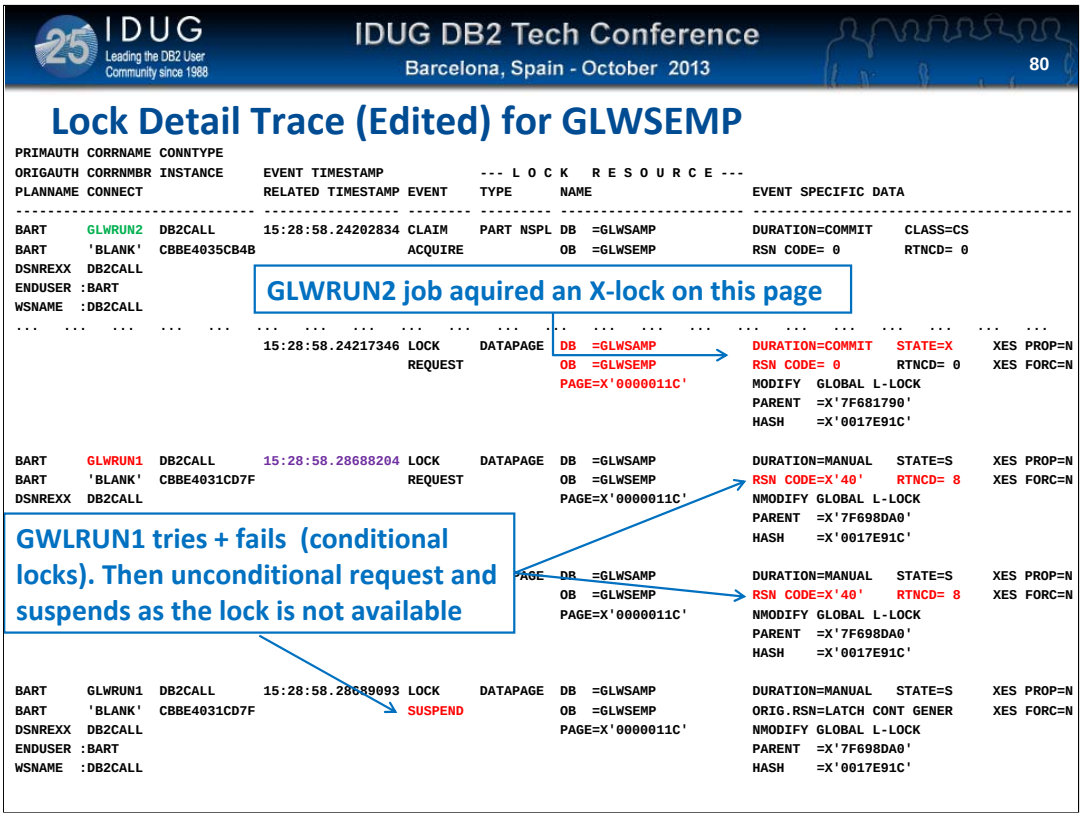
OMPE makes it convenient for you as it calculates the time the thd was suspended (SUSP.TIME) - that info is not part of the raw trace record data.

It also provides the timestamp the thd started its suspension so it is easy to find the point in the trace where we started to wait.

In this case there are no other suspensions in between the start and end of the suspension, but there you be many depending on the object's acvitity rate.

You can use the following //SYSIN DD * to create an OMPE lock suspension trace for a particular resouce (DB.TS/TB):

```
DB2PM
* ***********************************
*      GLOBAL PARMS
* ***********************************
 GLOBAL
      TIMEZONE (+4)
      INCLUDE( DB2ID(D1B1))
* ***********************************
*      LOCKING TRACE REPORT
***********************************
  LOCKING
         TRACE
          LEVEL(SUSPENSION)
          INCLUDE( DATABASE(GLWSAMP) PAGESET(GLWSEMP))
EXEC
```



The report has been edited to fit a bit better on the foil.

You can use the following //SYSIN DD * to create an OMPE lock detail trace for a particular resource (DB.TS/TB):

```
DB2PM
* ********************************
*       GLOBAL PARMS
* ********************************
 GLOBAL
      TIMEZONE (+4)
      INCLUDE( DB2ID(D1B1))
* ********************************
*       LOCKING TRACE REPORT
**********************************
  LOCKING
         TRACE
           LEVEL(DETAIL
           INCLUDE( DATABASE(GLWSAMP) PAGESET(GLWSEMP))
EXEC
```

IFCID 21 in a RECTRACE – Conditional Lock?

```
 LOCATION: DB1B                    OMEGAMON XE FOR DB2 PERFORMANCE EXPERT (V5R2M0)          PAGE: 1-34404
      GROUP: DB1BG                          RECORD TRACE - LONG                    REQUESTED FROM: NOT SPECIFIED
     MEMBER: D1B1                                                                            TO: NOT SPECIFIED
  SUBSYSTEM: D1B1
DB2 VERSION: V11
PRIMAUTH CONNECT   INSTANCE
ORIGAUTH CORRNAME  CONNTYPE
PLANNAME CORRNMBR
-------- -------- ----------
BART     DB2CALL  CBBE4031CD7
BART     GLWRUN1  DB2CALL     15:28:58.28688204 170869   1  21 LOCK DETAIL    NETWORKID:  USIBMSC    LUNAME:  SCPD1B1    LUWSEQ:    275
DSNREXX  'BLANK'                    17.73900397
         |-------------------------------------------------------------------------------------------------------------------------------
         |LOCK RES TYPE: DATA PAGE LOCK                        DBID: GLWSAMP          OBID: GLWSEMP   RESOURCE ID: X'0000011C00'
         |IRLM FUNC CODE : LOCK (NAME)          RETURN TOKEN: X'FFFFFFFF'             REQUEST TOKEN      : X'00000000'
         |LOCK STATE     : SHARED               DB2 TOKEN   : X'0099B0001E3FF428'     IRLM RETURN CODE   :          8
         |LOCK ATTRIBUTES: NMODIFY NOFORCE      PROP TO XES : NO                      ASYN TO XES        : NO
         |LOCK DURATION  : MANUAL               REQUEST TYPE:                         IRLM RETURN SUBCODE: B'0100000000000000'
         |PARENT TOKEN   : X'7F698DA0'          GLOBAL/LOCAL: GLOBAL                  OWNER              : 'BLANK'
         |CACHED STATE   : N/A                                                        LOCK HASH VALUE    : X'0017E91C'
         |QW0021CL: X'00'         QW0021U : X'012500401E754180'  QW0021FL: B'11110000'  QW0021CT: X'000004B6077CE501'
         |QW0021F3: B'00000000'   QW0021O : X'012500401E7540C0'  QW0021IR: X'0000'       QW0021F2: B'00000000'
         |-------------------------------------------------------------------------------------------------------------------------------
```

Return code	Reason code (byte 1)	Reason code (byte 2)	Description
08 (X'08')	X'80'		A non-recoverable system error occurred while processing this request.
	X'40'		The lock is not available in the requested state. MODE=COND was specified.

```
QW0021FL          REQUEST TYPE OR MODE:
QW0021CD            (S) X'80'  1 = CONDITIONAL          0 = UNCONDITIONAL
QW0021AQ            (S) X'40'  1 = AUTOMATIC RELEASE    0 = ACQUIRE
QW0021US            (S) X'20'  1 = SINGLE UNLOCK        0 = UNLOCK GROUP
QW0021SY            (S) X'10'  1 = RESULTANT STATE      0 = COUNT BY STATE
```