SQLCODE -911 Resolution Tips

SoftBase
A Fresche Solutions Brand

SQL error code -911 – one of the more difficult to debug

Determining the root cause of SQL code -911 deadlocks and deadly embraces typically is the task of DBAs. This involves a time-consuming search through various Db2 logs or z/OS console messages to determine who had the lock on a needed resource at the time the SQLCODE -911 was issued. Application Programmers usually don't have all the information available to determine the cause of the SQLCODE -911 and to resolve it themselves. Even if a programmer gets information from the SQLCA using DSNTIAR or something similar, only the name of the resource is known. The resource who held the lock that conflicted with the program's request is not readily evident.

Typical -911 Error

> DSNT408I SQLCODE = **-911,** ERROR: THE CURRENT UNIT OF WORK HAS BEEN ROLLED BACK
> DUE TO DEADLOCK OR TIMEOUT. REASON 00C9008E, TYPE OF RESOURCE 00000304, AND RESOURCE NAME TD
> GF6160.TDGF10 .X'000082' '.X'01' DSNT418I SQLSTATE = 40001 SQLSTATE RETURN CODE **00C9008E**

Db2 Messages Manual

> **00C9008E**
> Explanation: A lock request for the resource identified by NAME could not be granted, and the request waited for a period longer than the maximum specified by the installation.
> System action: The data manager function that detected this condition returns resource not available to its invoker.

Sometimes all that is needed to determine who has the lock is a display command of the space that held the lock. If the application developer has that authority, they can issue the command:

```
-DISPLAY DATABASE(TDGF6160) SPACENAM(TDGF10) LOCKS LIMIT(1000)
```

The results show that user CSBI has the resource locked for exclusive use:

```
DSNT360I  -DBCG **********************************
DSNT361I  -DBCG *  DISPLAY DATABASE SUMMARY
                *    GLOBAL LOCKS
DSNT360I  -DBCG **********************************
DSNT362I  -DBCG     DATABASE = TDGF6160  STATUS = RW
                  DBD LENGTH = 36332
DSNT397I  -DBCG
NAME     TYPE PART  STATUS            CONNID   CORRID       LOCKINFO
-------- ---- ----- ----------------- -------- ------------ ---------
TDGF10   TS         RW                TSO      CSBI         H-IX,S,C
    -                  AGENT TOKEN 174951
10       TB                           TSO      CSBI         H-IX,T,C
    -                  AGENT TOKEN 174951
******* DISPLAY OF DATABASE TDGF6160 ENDED      **********************
```

This time the developer is lucky. He can find the user who has the resource. Often, by the time he is locked out, the user who had the resource no longer has it. In fact, sometimes it can be another user or job that has the resource locked by the time the display command is issued.

SQLCODE -911 Resolution Tips

SoftBase
A Fresche Solutions Brand

What is Needed

The application programmer needs to know who held the resource that their application wants. Further, the job transaction, etc who held the interfering lock needs to be notified that their resource was held longer than the time specified in DSNZPARM IRLMRWT.

Typical Causes of -911

- Lock escalation in the process holding the lock. Perhaps the limit in number of locks per tablespace NUMLKTS was exceeded causing lock escalation
- Inappropriate tablespace locksize. Locksize page or tablespace when locksize row was needed etc.
- Inappropriate bind isolation level in the process holding the lock. Bind isolation RR instead of CS for example
- Use of LOCK TABLE statement in the process holding the lock
- Referential integrity constraints without an index on the foreign key in the process holding the lock.
  Delete of a parent row is checking the restrict constraint for its children but there is no index in the child to support the check.
- Insufficient Commit Frequency in the process holding the lock
- Bad access path in the process holding the lock
- Testing a program with step debugger– the process holding the lock

In all but one of the typical causes above, ***the issue is in the process that holds the lock rather than the process who gets the -911***. In normal DB2 messaging, the only process that is aware of the problem is the one receiving the -911. Both the process who holds the lock and the process receiving the deadlock -911 SQLCODE needs to know who the lock holder was at the time of the deadlock. Perhaps this holder of the lock is a scheduled process that wants the lock for an extended period. More often the problem needs to be addressed by the holder of the resource.

SoftBase's DeadLock Advisor eliminates the legwork by revealing the offender and the victim of SQLCODE -911 by placing a message in each respective joblog. See: https://www.softbase.com/pdf/datasheets/Db2_Deadlock_Advisor.pdf

In conclusion, Db2 deadlocks and timeouts require investigation from both the cause and the victim perspective. Any changes that are made should be closely monitored to insure the contention is resolved and no other complications are introduced.