

# SOLVING CONCURRENT AND DEAD LOCK PROBLEMS AT DATABASE LEVEL

Bostan Constantin

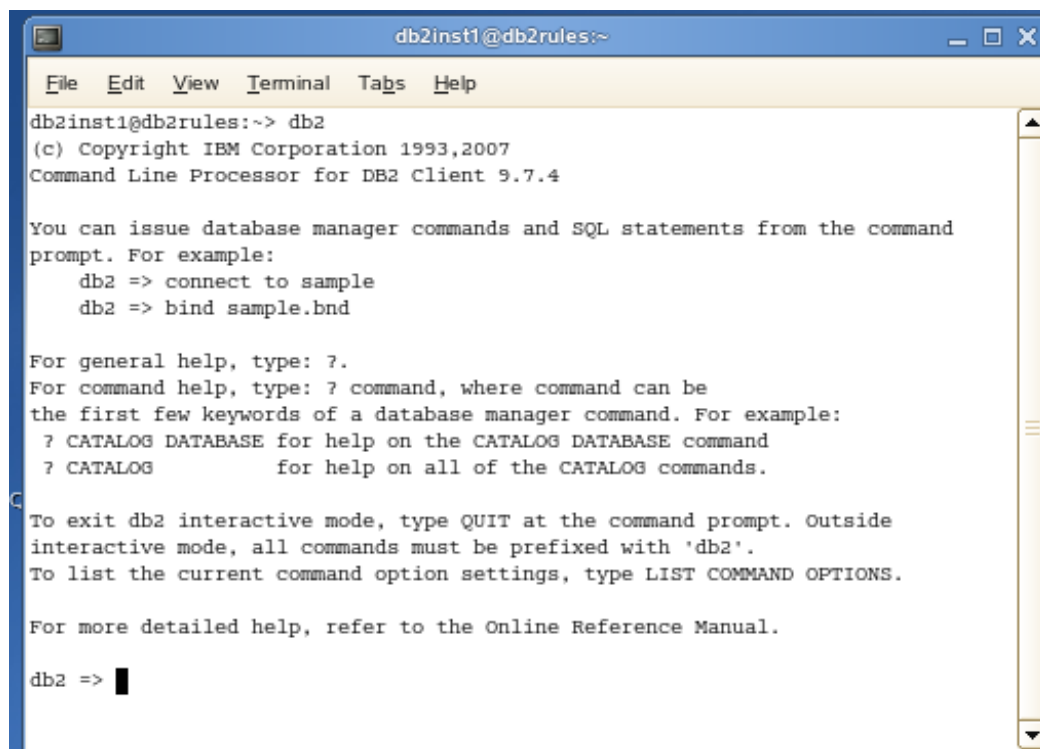
Technical University of Moldova

**Abstract:** The main problem in having a program that runs on many computers and interacts with one database is to secure the execution of a query. The software used to solve these concurrent problems is DB2 (Database2), it is also distributed free and can be used by everyone. And for Java users there is an Eclipse plug-in called Data Studio. I have formulated 4 problems that could occur in SQL transactions and their solutions in DB2 by configuring the server and queries with specific options. I have also described how to solve the dead lock problem with some specific server configurations.

**Keywords:** DataBase2, Eclipse, Concurrency, Dead-Lock, SQL, Data Studio, Eclipse.

## 1. Introduction to IBM Database2 and Data Studio

DB2 (Database2) is a database management system responsible to store and process information. It is a free tool that can be downloaded from official IBM web site. It has no user and database size limit. We can access the created database using command line (in windows) or terminal (in Linux) as shown in figure 1.



```
db2inst1@db2rules:~  
File Edit View Terminal Tabs Help  
db2inst1@db2rules:~> db2  
(c) Copyright IBM Corporation 1993,2007  
Command Line Processor for DB2 Client 9.7.4  
  
You can issue database manager commands and SQL statements from the command  
prompt. For example:  
  db2 => connect to sample  
  db2 => bind sample.bnd  
  
For general help, type: ?.  
For command help, type: ? command, where command can be  
the first few keywords of a database manager command. For example:  
 ? CATALOG DATABASE for help on the CATALOG DATABASE command  
 ? CATALOG           for help on all of the CATALOG commands.  
  
To exit db2 interactive mode, type QUIT at the command prompt. Outside  
interactive mode, all commands must be prefixed with 'db2'.  
To list the current command option settings, type LIST COMMAND OPTIONS.  
  
For more detailed help, refer to the Online Reference Manual.  
  
db2 => █
```

Figure 1- Terminal in Linux that runs DB2

Usual to enter a command we start with keyword db2 <command to execute>. The typical commands are executing SQL queries or configuring the server with command: db2 get db for <database\_name>

Database studio is responsible for managing data for DB2. It is also a free tool that can be installed in Eclipse. The program is shown in figure 2. Database2 can be used with many programming language such as: Java, C/C++, C#, Python, etc. Therefore we can develop different programs that will interact with the same database.

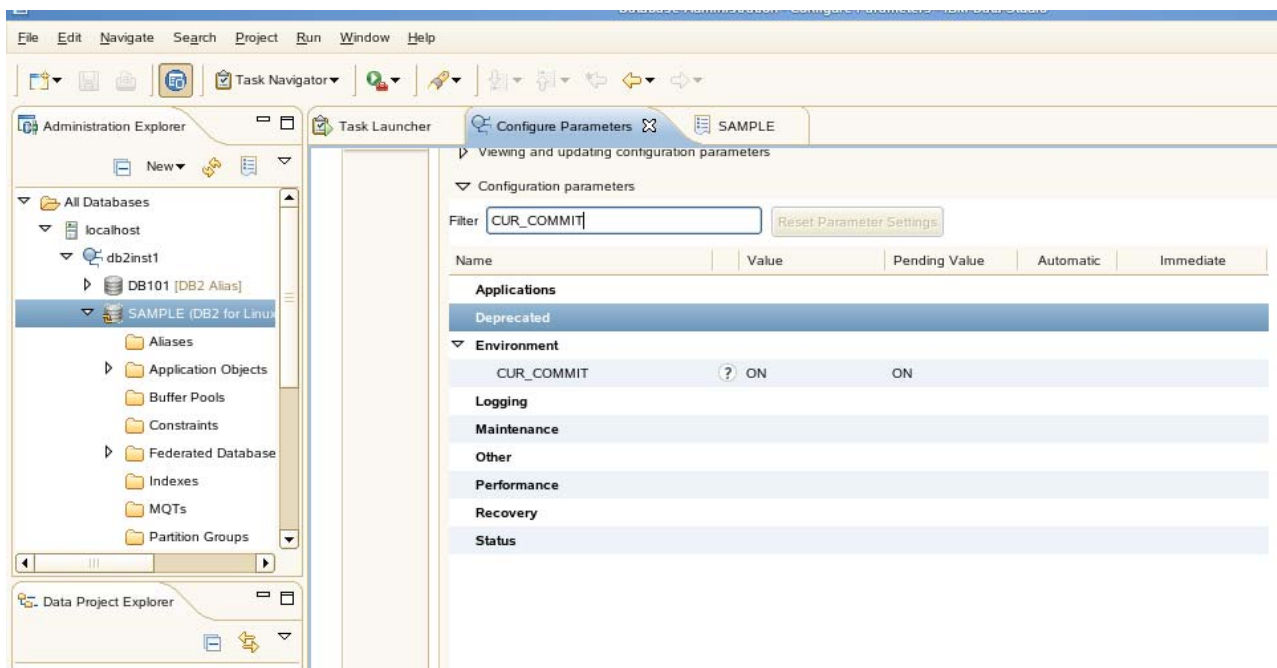


Figure 1- Configuring the Server using Data Studio in Eclipse

## 2. Formulating the problem.

In each database we have transactions that usually are SQL operations: SELECT, UPDATE, DELETE, and INSERT. Transactions end up with statements like: COMMIT and ROLLBACK. COMMIT statement changes permanent the database and ROLLBACK is reversing the changes. These two statements release all the locks of an SQL query transaction, so that other operation could take place.

We have 4 rules of transaction named ACID rules:

- 1) Atomicity: all statements in the transaction are treated and executed as one unit;
- 2) Consistency: Any transaction will take the data from one consistent state to another, so only valid consistent data is stored in the database;
- 3) Isolation: concurrent transaction cannot interfere with each other;
- 4) Durability: if a change is made it persist in the database.

The main problem in concurrent programming is accessing the same resources at the same time by multiple users or threads. The locking mechanism will ensure data integrity atomicity of a transaction.

I have formulated 4 concurrent problems that could occur in a database transaction.

- 1) Lost update: When two threads are updating concurrently the same record breaking the atomicity rule.
- 2) Uncommitted read: the lost data of an SQL select query caused by ROLLBACK statement. For example we have two threads that are accessing concurrently the same database, thread 1 is updating the database, thread 2 is reading data, at one moment thread 1 rolls back the transaction causing thread 2 to consider that there is data;
- 3) Non repeatable read: When the same select query give you different result because in the background a thread is changing the database;
- 4) Phantom read: appears when we have data fields with null and non null values and one thread is setting all data to null, performing delete operation and another thread is performing the select operation giving two different results: one that have data and another result that outputs only null values.

## 3. Solving the problem

IBM DB2 solves this problem by using isolation levels concept which locks some operations on the Database if there is a transaction. The isolation levels are:

- 1) Uncommitted Read (UR)
- 2) Cursor Stability (CS)
- 3) Currently committed (CC)
- 4) Read Stability (RS)
- 5) Repeatable Read (RR)

In table 1 it is specified what kind of problem each isolation level solves. As we see the most useful is RR (Repeatable Read) but it has a high CPU cost. The least costless and easy to implement is UR (Uncommitted

Read) that could secure only for Lost Update and it doesn't give you the chance of rolling back the transaction.

Table1. Isolation level

	Lost Update	Uncommitted read	Non-Repeatable read	Phantom Read
Repeatable Read (RR)	-	-	-	-
Read Stability (RS)	-	-	-	Possible
Cursor Stability (CS)	-	-	Possible	Possible
Uncommitted Read (UR)	-	Possible	Possible	Possible

To specify in an SQL query to use a specific isolation level we use WITH {RR, RS, CS, UR} clauses for example:

```
SELECT * FROM TableName WITH UR
```

Also we can configure the server using the command

```
cur_commit db cfg parameter to enable/disable
```

The Enabled configuration Currently Commit gives you the possibility to block the reader to select information from database, this is shown in figure 3.

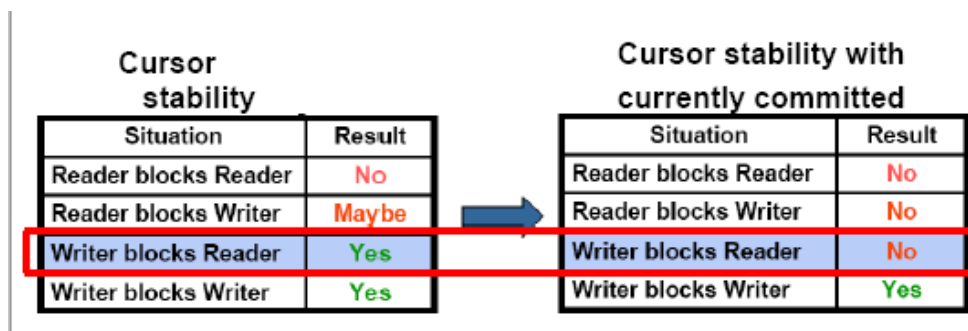


Figure 3- Operations that can be performed by enabling Currently Committed

#### 4. Dead lock problem

Dead lock appears when two threads or more threads wait indefinitely for a resource. Each application is holding a resource the other needs. And these resources are never released causing a dead lock of these two programs. For example we have two threads that run multiple transactions with disabled Currently Committed which gives the possibility to Writer to block the Reader. As the result if these two threads are running some update operation on one database they will block each other indefinitely, this result is shown in figure 3.

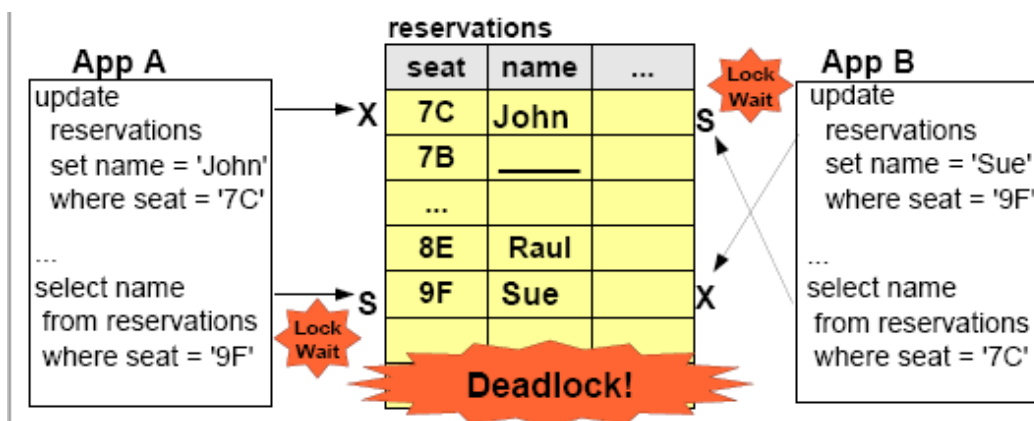


Figure 3- Example of dead lock in a transaction

### **5. Solving the Dead lock problem**

By default DB2 waits indefinitely to obtain any needed result. The LOCKTIMEOUT specifies the number of seconds to wait for a lock to be released. The default value is -1 which means to wait infinite for the result. The other solution is to use DLCHKTIME to set the time interval for checking deadlocks. DB2 uses an internal mechanism to select which transaction to rollback and which one to continue. If a transaction is rolled back it releases all the locks. LOCKTIMEOUT and DLCHKTIME are configured using DB2 cfg command.

### **Bibliography**

1. Neeraj S., Perniu L., Raul F. Chong, Abhishek I., Nandan C. *Database Fundamentals*. IBM Canada, 2010.
2. Faeth A., Bhatia D., Zilio D. *IBM Data studio for DB2*. IBM Canada, 2012, p.7-63.
3. IBM Corporation. *Data concurrency and loking*. PDF File Presentaion [www.coursehero.com/file/7014336/15-Data-Concurrency-and-Locking](http://www.coursehero.com/file/7014336/15-Data-Concurrency-and-Locking), p.1-76.