Resolving Locks and Concurrency Issues

You will need to open 3 sessions for this Practice and work in parallel for each Case

SESSION 1 -- **user SYSTEM** (here our Junior DBA will act as a plain User)

```
SQL> show user
USER is "SYSTEM"

SQL> set pagesize 100
SQL>  SELECT empno, ename, sal FROM scott.emp
        WHERE ename IN ('JAMES','ADAMS');

EMPNO    ENAME          SAL
----------    ----------      ----------

     7900    JAMES          950

     7876    ADAMS         1100

2 rows selected.
```

CASE 1 -- CONCURRENCY AND EXCLUSIVE ROW LOCKS (DIFFERENT ONES)
==

```
SQL> UPDATE scott.emp
  2  SET    sal = 1000
  3  WHERE  ename = 'JAMES';

1 row updated.

SQL> SELECT empno, ename, sal FROM scott.emp
    WHERE ename IN ('JAMES','ADAMS');

EMPNO    ENAME             SAL
----------    ----------         ----------

     7900    JAMES             1000

     7876   ADAMS              1100
```

* Well, SYSTEM does NOT see the new value for SCOTT's update, but can see his new update *

```
SQL> ROLLBACK;
Rollback complete.
```

CASE 2 -- CONCURRENCY AND EXCLUSIVE ROW LOCKS (SAME ROW)
==

```
SQL> UPDATE scott.emp
  2  SET    sal = 1000
  3  WHERE  ename = 'JAMES';
1 row updated.
```

* Here SYSTEM will go for a donut (but without Commit or Rollback) *

```
SQL> DESC dba_objects
ERROR:
ORA-03135: connection lost contact
```

* After SYSTEM came back from the Donut shop he discovered his session was terminated by SYSDBA, meaning an AUTO-ROLLBACK followed by Server *

CASE 3 -- DEADLOCK
===================

```
SQL> conn system
Enter password:
Connected.

SQL> set pagesize 100

SQL> UPDATE scott.emp
  2  SET    sal = 1000
  3  WHERE  ename = 'JAMES';

1 row updated.

SQL> SELECT ename, sal FROM scott.emp WHERE ename IN ('JAMES','ADAMS');

ENAME             SAL
----------  ----------
JAMES            1000
ADAMS            1100

SQL> UPDATE scott.emp
  2  SET    sal = 1000
  3  WHERE  ename = 'ADAMS';
UPDATE scott.emp
       *
ERROR at line 1:
ORA-00060: deadlock detected while waiting for resource
```

* Server will AUTO-DETECT the DEADLOCK and Rollback just the last DML attempt by the first user (here it was SYSTEM) *

```
SQL> SELECT ename, sal FROM scott.emp
        WHERE ename IN ('JAMES','ADAMS');

ENAME           SAL
----------  ----------
JAMES          1000
ADAMS          1100
```

* So, only the SECOND update was auto-rollbacked and SYSTEM still needs to decide about the first update *

```
SQL> ROLLBACK;

Rollback complete.
```

SESSION 2 -- user SCOTT (our regular user)

```
SQL> show user
USER is "SCOTT"
```

CASE 1 -- CONCURRENCY AND EXCLUSIVE ROW LOCKS (DIFFERENT ONES)
==

```
SQL> SET PAGESIZE 100
SQL> SELECT empno, ename, sal FROM scott.emp
        WHERE ename IN ('JAMES','ADAMS');

EMPNO  ENAME           SAL
---------- ----------      ----------

     7900  JAMES             950

     7876  ADAMS            1100

2 rows selected.

SQL> UPDATE emp
  2  SET    sal = 1200
  3  WHERE  ename = 'ADAMS';  --> attempting to update a different row from user System
                                  and NO WAIT will happen
1 row updated.
```

```
SQL> SELECT empno, ename, sal FROM scott.emp
        WHERE ename IN ('JAMES','ADAMS');

EMPNO  ENAME          SAL
---------- ------------       ----------

     7900  JAMES             950

     7876  ADAMS          1200
```

* Well, SCOTT does NOT see the new value for SYSTEM's update, but can see his new update *

```
SQL> ROLLBACK;

Rollback complete.
```

CASE 2 -- CONCURRENCY AND EXCLUSIVE ROW LOCKS (SAME ROW)
==

```
SQL> UPDATE emp
  2  SET    sal = 1200
  3  WHERE  ename = 'JAMES';    --> attempting to update row already being updated by
                                    user  System
1 row updated.
```

* HERE THIS SESSION WAITS TILL USER SYSTEM (Blocking Session) was killed by DBA. Look at the end of this Practice for EM situation and how to handle it there *

```
SQL> ROLLBACK;

Rollback complete.
```

CASE 3 -- DEADLOCK
====================

```
SQL> UPDATE emp
  2  SET    sal = 1200
  3  WHERE  ename = 'ADAMS';

1 row updated.

SQL> SELECT ename, sal FROM scott.emp
       WHERE ename IN ('JAMES','ADAMS');
```

```
ENAME           SAL
----------      ----------
JAMES            950
ADAMS           1200

SQL> UPDATE emp
  2  SET    sal = 1200
  3  WHERE  ename = 'JAMES';
```

* Here SCOTT will wait just for a second (or two) before Server will AUTO-DETECT the DEADLOCK and Rollback the last DML attempt by the first user (SYSTEM) *

```
1 row updated.

SQL> SELECTename, sal FROM scott.emp
        WHERE ename IN ('JAMES','ADAMS');

    ENAME           SAL
    ------------    ----------

    JAMES           1200

    ADAMS           1200

SQL> ROLLBACK;

Rollback complete.
```

SESSION 3 -- MONITORING by SYSDBA

```
SQL> show user
USER is "SYS"

SQL> SET  PAGESIZE 100
```

CASE 1 -- CONCURRENCY AND EXCLUSIVE ROW LOCKS (DIFFERENT ONES)
===

```
SQL> SELECT sid, serial#, username
  2      FROM   V$SESSION WHERE username IS NOT NULL;

       SID    SERIAL#    USERNAME
----------    ----------    ------------------------------
```

```
        134            95    SYSMAN
        135           104    DBSNMP
        138             2   SYSMAN
        141           121   SCOTT    -- Session 2
        142             2   SYSMAN
        143           215   SYSTEM   -- Session 1
        145             2   SYSMAN
        149            15   DBSNMP
        152          1302   SYS
        159             3   SYS

10 rows selected.

SQL> SELECT  sid, type, id1, lmode, request
  2      FROM    V$LOCK WHERE  type IN ('TM','TX');

       SID TY        ID1      LMODE    REQUEST
---------- ----   ---------      ----------     ---------
       143 TM      11848          3           0
       141 TM      11848          3           0
       141 TX     458771         6           0
       143 TX      65558          6           0
```

* Well, both SYSTEM and SCOTT are holding a Share Table Lock (Mode 3) on table SCOTT.EMP (TM), while holding EXCLUSIVE ROW LOCK (Mode 6) on different rows (TX) and that is why NO lock request exists (NOBODY is waiting) *

```
SQL> DESC dba_objects
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 OWNER                                              VARCHAR2(30)
 OBJECT_NAME                                        VARCHAR2(128)
 SUBOBJECT_NAME                                     VARCHAR2(30)
 OBJECT_ID                                          NUMBER
 DATA_OBJECT_ID                                     NUMBER
 OBJECT_TYPE                                        VARCHAR2(19)
 CREATED                                            DATE
 LAST_DDL_TIME                                      DATE
 TIMESTAMP                                          VARCHAR2(19)
 STATUS                                             VARCHAR2(7)
 TEMPORARY                                          VARCHAR2(1)
 GENERATED                                          VARCHAR2(1)
 SECONDARY                                          VARCHAR2(1)

SQL> SELECT owner, object_name, object_type
         FROM   dba_objects WHERE  object_id = 11848;
```

```
OWNER
------------------------------
OBJECT_NAME
--------------------------------------------------------------------------------
OBJECT_TYPE
-------------------
SCOTT
EMP
TABLE
```

CASE 2 -- CONCURRENCY AND EXCLUSIVE ROW LOCKS (SAME ROW)
==

```
SQL> SELECT  sid, type, id1, lmode, request
        FROM    V$LOCK WHERE  type IN ('TM','TX');

      SID TY        ID1      LMODE    REQUEST
---------- ----   ----------   ----------   ----------
      141 TX     393257          0          6
      141 TM      11848          3          0
      143 TM      11848          3          0
      143 TX     393257          6          0
```

* Well, it is clear WHO is blocking and WHO is waiting now. Request is made for the same row FROM session 141 (Waiting) and that row is held (locked) by session 143 (Blocking). LOCK MODE BYTE is set on session 9 (value 6) and NOT set for session 10 (value 0), SYSDBA will terminate blocking session 143 for user SYSTEM *

```
SQL> ALTER SYSTEM KILL SESSION '143,215' IMMEDIATE;

System altered.
```

CASE 3 -- DEADLOCK
===================

* In the DEADLOCK case, DBA needs to do NOTHING, because Server will AUTO-DETECT it and rollback the last DML update by the first user (here SYSTEM session). DBA may check the ALERT LOG FILE where the entry about the ORA-0060 error can be found and a reference to the USER TRACE file where the details about the DEADLOCK can be analyzed *

```
SQL> HOST
db091a32@dbaoracle3:~> pwd
/home/db091a32

db101a32@dbaoracle3:~> cd oradata
db101a32@dbaoracle3:~/oradata> ls -l
```

```
total 12
drwxrwx--- 3 oracle db091a32 4096 Jan 17 21:19  admin
drwxrwx--- 2 oracle db091a32 4096 Jan 17 11:22  db101a32
-rw-rw---- 1 oracle db091a32 2560   Jan 17 21:37  spfiledb101a32.ora

db091a32@dbaoracle3:~/oradata> cd admin
db091a32@dbaoracle3:~/oradata/admin> ls -l
total 4
drwxrwx--- 6 oracle db091a32 4096 Dec 17 21:19 db101a32

db091a32@dbaoracle3:~/oradata/admin> cd db091a32/bdump
db091a32@dbaoracle3:~/oradata/admin/db091a32/bdump> ls -l al*
-rw-rw---- 1 oracle db091a32 218349 Feb 19 11:23 alert_db091a32.log

db101a32@dbaoracle3:~/oradata/admin/db101a32/bdump> tail -15 alert*

Thread 1 advanced to log sequence 668
  Current log# 2 seq# 668 mem# 0: /home/db091a32/oradata/db091a32/redo02.log
Thread 1 cannot allocate new log, sequence 669
Checkpoint not complete
  Current log# 2 seq# 668 mem# 0: /home/db091a32/oradata/db091a32/redo02.log
Thu Feb 19 11:00:15 2009
Thread 1 advanced to log sequence 669
  Current log# 3 seq# 669 mem# 0: /home/db091a32/oradata/db091a32/redo03.log
Thu Feb 19 11:16:03 2009
Immediate Kill Session#: 143, Serial#: 215
Immediate Kill Session: sess: 0x2fef90d4  OS pid: 7184
Thu Feb 19 11:21:30 2009
Thread 1 advanced to log sequence 670
  Current log# 1 seq# 670 mem# 0: /home/db091a32/oradata/db091a32/redo01.log
Thu Feb 19 11:23:23 2009
```

ORA-00060: Deadlock detected. More info in file /home/db101a32/oradata/admin/db091a32/udump/db101a32_ora_16947.trc.

```
db101a32@dbaoracle3:~/oradata/admin/db091a32/bdump> cd ../udump

db091a32@dbaoracle3:~/oradata/admin/db091a32/udump> head -40 *16947*

/home/db101a32/oradata/admin/db101a32/udump/db091a32_ora_16947.trc
Oracle Database 10g Enterprise Edition Release 10.2.0.1.0 - Production
With the Partitioning, OLAP and Data Mining options
ORACLE_HOME = /opt/oracle/10.2.0.1.0
System name:    Linux
Node name:      dbaoracle3
Release:        2.6.22.17-0.1-bigsmp
Version:        #1 SMP 2008/02/10 20:01:04 UTC
```

```
Machine:       i686
Instance name: db091a32
Redo thread mounted by this instance: 1
Oracle process number: 15
Unix process pid: 16947, image: oracle@dbaoracle3 (TNS V1-V3)

*** 2009-02-19 11:23:23.126
*** ACTION NAME:() 2009-02-19 11:23:23.126
*** MODULE NAME:(SQL*Plus) 2009-02-19 11:23:23.126
*** SERVICE NAME:(SYS$USERS) 2009-02-19 11:23:23.126
*** SESSION ID:(140.842) 2009-02-19 11:23:23.126
DEADLOCK DETECTED
[Transaction Deadlock]
Current SQL statement for this session:
UPDATE scott.emp
SET    sal = 1000
WHERE  ename = 'ADAMS'
The following deadlock is not an ORACLE error. It is a
deadlock due to user error in the design of an application
or from issuing incorrect ad-hoc SQL. The following
information may aid in determining the deadlock:
Deadlock graph:
                       ---------Blocker(s)--------  ---------Waiter(s)---------
Resource Name          process session holds waits  process session holds waits
TX-00050014-00000179       15     140     X             25     141          X
TX-00010016-0000017c       25     141     X             15     140          X
session 140: DID 0001-000F-0000002E     session 141: DID 0001-0019-00000003
session 141: DID 0001-0019-00000003     session 140: DID 0001-000F-0000002E
Rows waited on:
Session 141: obj - rowid = 00002E48 - AAAC5IAAEAAAAAfAAH
  (dictionary objn - 11848, file - 4, block - 31, slot - 7)
Session 140: obj - rowid = 00002E48 - AAAC5IAAEAAAAAfAAM
```

Resolving Locking Conflict in EM Cloud Control (Case 2)

After getting a phone call from user SCOTT that his update is hanging (being frozen), SYSDBA will investigate the Blocking / Waiting graph and will kill (terminate) the Blocking session of user SYSTEM (who went for donut without ending his transaction).

In EM Cloud Control from Home Database Page do the following:
Performance →Blocking Sessions and you will get this page:

Blocking Sessions

Page Refreshed Feb 19 11:04:22 AM

View Session

Expand All | Collapse All

Select	Username	Sessions Blocked	Session ID	Session Serial Number	SQL Hash Value	Wait Class	Wait Event	P1	P2	I
○	Blocking Sessions									
●	SYSTEM	1	143	215		Idle	SQL*Net message from client	1650815232	1	0
○	SCOTT	0	141	121	1zr96qffmycx8	Application	enq: TX - row lock contention	1415053318	393257	3

Here is obvious that user SYSTEM has blocked user SCOTT and then you will select SYSTEM session and click View Session and then you get the following page, where you will click on Previous SQL link

Session Details: SYSTEM (143)

Collected From Target Feb 19, 2009 11:13:07 AM

View Data Real Time: 15 Second Refresh Refresh

Kill Session Enable SQL Trace Disable SQL Trace

General Activity Statistics Open Cursors Blocking Tree Wait Event History

Server		Client		Application	
Current Status	INACTIVE	OS User Name	db091a32	Current SQL	None
Serial Number	215	OS Process ID	6635	Current SQL Command	UNKNOWN
DB User Name	SYSTEM	Host	dbaoracle3	**Previous SQL**	3qx6vry62w623
OS Process ID	7184	Terminal	pts/4	Last Call Elapsed Time	12 Minutes, 53 Seconds
Logged On Since	Feb 19, 2009 10:42:03 AM	Current Client ID	Unavailable	SQL Trace	DISABLED
				Open Cursors	24
				Program	sqlplus@dbaoracl

Logged On For	31 Minutes, 4 Seconds	Current Client Info	Unavailable		e3 (TNS V1-V3)
Connection Type	DEDICATED			Service	SYS$USERS
Type	USER			Current Module	SQL*Plus
Resource Consumer Group	Unavailable			Current Action	Unavailable

Contention

Blocking Session ID	None

Wait

Current Wait Event	SQL*Net message from client
Current Wait Class	Idle
Waiting for	12 Minutes, 53 Seconds
P1	driver id 1650815232
P2	#bytes 1
P3	None
Object	None

Now you will see the BLOCKING STATEMENT and also you can see the Explain Plan (if you click on Plan link)

Text

```
UPDATE scott.emp
SET sal = 1000
WHERE ename = 'JAMES'
```

Details

Select the plan hash value to see the details below. Plan Hash Value 1494045816

Statistics Activity Plan Tuning Information

Now you need to return to the Blocking Session page by using ← Option of your browser and then select SCOTT session and then click on its SQL Hash value. Then you will see what is SCOTT attempting to do (the Waiting session). Here is given just the SQL statement and you can conclude that these two users are trying to update the same row (for employee James).

Blocking Sessions

Page Refreshed Feb 19, 2009 11:10:27 AM

View Session Kil

Select	Username	Sessions Blocked	Session ID	Session Serial Number	SQL Hash Value	Wait Class	Wait Event	P1	P2	P
○	▼ Blocking Sessions									
○	▼ SYSTEM	1	143	215		Idle	SQL*Net message from client	1650815232	1	0
●	SCOTT	0	141	121	1zr96qffmycx8	Application	enq: TX - row lock contention	1415053318	393257	3

SQL Details: 1zr96qffmycx8
Text

```
UPDATE scott.emp
SET sal = 1200
WHERE ename = 'JAMES'
```

You will need to terminate your Blocking session by selecting SYSTEM session and click on Kill Session

Are you sure you want to kill this session?

SID 143

DB User SYSTEM

Program sqlplus@dbaoracle3 (TNS V1-V3)

Options [✓ Kill Immediate] Post Transactional

```
ALTER SYSTEM KILL SESSION '143,215' IMMEDIATE
```

After this session was terminated you need to Refresh your browser and then you will get:

Blocking Sessions

Page Refreshed Feb 24, 2006 9:18:50 PM

Select	Username	Sessions Blocked	Session ID	Session Serial Number	SQL Hash Value	Wait Class	Wait Event	P1	P2	P3	Seconds in Wait
	No sessions found to be currently blocking other sessions.										

Finally, user SCOTT will finish his Update, because the initial one by SYSTEM was roll backed (after SYSDBA killed his session) and you can see in SCOTT's session that he got his SQL prompt back.