Note *: This is a slightly old (year 2007) document on a test case showing how an INSERT can be waiting on a lock.*

Sesssion 43 (text in GREEN), which is inserting the record with the value "holder_s1" is the session that HOLDs the "lock" that is present if there is a Unique Key for the row being inserted

Session 45 (text in ORANGE), which is inserting the record with the value "waiter_s2" is the session that is attempting to insert a duplicate row on the Unique Key before the Holder has commited and is, therefore, the WAITER

Session 55 (text in BLUE) is the third session that I use to run queries on V$LOCK, V$TRANSACTION, V$SESSION_WAIT and V$SESSION to show transactions, locks, holders and waiters.

Setup the test table and the first “Insert Transaction”.

```
20:33:41 SQL>
20:33:41 SQL> drop table my_test_table;

Table dropped.

20:33:41 SQL>
20:33:41 SQL> create table my_test_table (col1 number , col2  varchar2(12)) ;

Table created.

20:33:41 SQL>
20:33:41 SQL> REM See if there locks on a non-unique insert
20:33:41 SQL>
20:33:41 SQL> insert into my_test_table values (1,'holder_s1');

1 row created.

20:33:41 SQL>
20:33:41 SQL> pause Press ENTER to proceed
Press ENTER to proceed
```

The “transaction” hasn’t committed. Let’s see what we find on Locks and Transactions.

```
20:33:52 SQL>
20:33:52 SQL> pause Current Lock Holders on MY_TEST_TABLE
Current Lock Holders on MY_TEST_TABLE

20:33:54 SQL> @Lock_Holder MY_TEST_TABLE  HEMANT
20:33:54 SQL> set pages 100
20:33:54 SQL> set lines 100
20:33:54 SQL> set verify off
20:33:54 SQL>
20:33:54 SQL> col c for a20 heading 'Object'
20:33:54 SQL> col b format a20 heading 'Lock Type'
20:33:54 SQL> col d format a11 heading 'Lock Held'
20:33:54 SQL> col e format a11 heading 'Lock Req.'
20:33:54 SQL> col f for 999999 heading 'Oracle SID'
20:33:54 SQL>
20:33:54 SQL> select  object c,
20:33:54   2           decode(w.type,
20:33:54   3                  'MR', 'Media Recovery',
20:33:54   4                  'RT', 'Redo Thread',
```

```
20:33:54   5                  'UN', 'User Name',
20:33:54   6                  'TX', 'Transaction',
20:33:54   7                  'TM', 'DML',
20:33:54   8                  'UL', 'PL/SQL User Lock',
20:33:54   9                  'DX', 'Distributed Xaction',
20:33:54  10                  'CF', 'Control File',
20:33:54  11                  'IS', 'Instance State',
20:33:54  12                  'FS', 'File Set',
20:33:54  13                  'IR', 'Instance Recovery',
20:33:54  14                  'ST', 'Disk Space Transaction',
20:33:54  15                  'TS', 'Temp Segment',
20:33:54  16                  'IV', 'Library Cache Invalidation',
20:33:54  17                  'LS', 'Log Start or Switch',
20:33:54  18                  'RW', 'Row Wait',
20:33:54  19                  'SQ', 'Sequence Number',
20:33:54  20                  'TE', 'Extend Table',
20:33:54  21                  'TT', 'Temp Table',
20:33:54  22                  w.type)  b ,
20:33:54  23            decode(lmode,
20:33:54  24                  0, 'None',     /* Mon Lock equivalent */
20:33:54  25                  1, 'Null',     /* N */
20:33:54  26                  2, 'Row-S (SS)',     /* L */
20:33:54  27                  3, 'Row-X (SX)',     /* R */
20:33:54  28                  4, 'Share',    /* S */
20:33:54  29                  5, 'S/Row-X (SSX)',  /* C */
20:33:54  30                  6, 'Exclusive',      /* X */
20:33:54  31                  to_char(lmode)) d ,
20:33:54  32             decode(request,
20:33:54  33                  0, 'None',     /* Mon Lock equivalent */
20:33:54  34                  1, 'Null',     /* N */
20:33:54  35                  2, 'Row-S (SS)',     /* L */
20:33:54  36                  3, 'Row-X (SX)',     /* R */
20:33:54  37                  4, 'Share',    /* S */
20:33:54  38                  5, 'S/Row-X (SSX)',  /* C */
20:33:54  39                  6, 'Exclusive',      /* X */
20:33:54  40                  to_char(request))  e ,
20:33:54  41                  w.id1,
20:33:54  42                  w.id2,
20:33:54  43                  x.sid f
20:33:54  44  from v$session x , v$access y ,  v$lock w
20:33:54  45  where x.sid = y.sid
20:33:54  46  and w.sid = y.sid
20:33:54  47  and object = upper('&1')
20:33:54  48  and owner = upper('&2')
20:33:54  49  order by d,  b , c, e;

no rows selected

20:33:54 SQL>
20:33:54 SQL> pause Current Active Transactions
Current Active Transactions

20:33:55 SQL> @active_transactions
20:33:55 SQL> set pages60
20:33:55 SQL> REM  select s.sid, s.serial#, p.spid, s.username, s.program,
20:33:55 SQL> REM  t.xidusn, t.used_ublk, t.used_urec, sa.sql_text from
20:33:55 SQL> REM  v$process p,v$session s, v$sqlarea sa, v$transaction t
```

```
20:33:55 SQL> REM  where s.paddr=p.addr
20:33:55 SQL> REM  and s.taddr=t.addr
20:33:55 SQL> REM  and s.sql_address=sa.address(+)
20:33:55 SQL> REM  and s.sql_hash_value=sa.hash_value(+)
20:33:55 SQL> REM  order by s.sid
20:33:55 SQL> select s.sid, s.serial#, p.spid, s.username, s.program,
20:33:55  2 t.xidusn, t.used_ublk, t.used_urec, s.last_call_et, sa.sql_text from
20:33:55   3  v$process p,v$session s, v$sqlarea sa, v$transaction t
20:33:55   4  where s.paddr=p.addr
20:33:55   5  and s.taddr=t.addr
20:33:55   6  and
decode(s.sql_address,'00',s.prev_sql_addr,s.sql_address)=sa.address
20:33:55   7  and
decode(s.sql_hash_value,'0',s.prev_hash_value,null,s.prev_hash_value,s.sql_ha
sh_value)=sa.hash_value
20:33:55   8  order by s.sid
20:33:55   9  /

       SID    SERIAL# SPID         USERNAME  PROGRAM          XIDUSN  USED_UBLK
USED_UREC
---------- ---------- ------------ --------- ------------ ---------- ----------
----------
LAST_CALL_ET
------------
SQL_TEXT
------------------------------------------------------------------------------
------------------------
        43         34 10420        HEMANT    sqlplus.exe           2          1
1
          12
insert into my_test_table values (1,'holder_s1')


20:33:55 SQL>
20:33:55 SQL>
20:33:55 SQL> pause  Sessions currently Waiting on an Enqueue
Sessions currently Waiting on an Enqueue

20:33:56 SQL> select sid, event, seconds_in_wait from v$session_wait where event
like 'enq%' order by sid;

no rows selected

20:33:56 SQL>
20:33:56 SQL> pause Waiting and blocking session from v$session
Waiting and blocking session from v$session

20:33:57 SQL> select sid, blocking_session from v$session where username =
'HEMANT';

       SID BLOCKING_SESSION
---------- ----------------
        43
        45
        55

20:33:57 SQL>
```

The first transaction (session 43) hasn't committed, does have an active transaction but doesn't appear to hold any locks.

Let's see what happens when session 45 attempts a transaction.

```
20:34:12 SQL>
20:34:12 SQL>
20:34:12 SQL> REM See if there we wait on locks on a non-unique insert
20:34:12 SQL>
20:34:12 SQL> insert into my_test_table values (1,'waiter_s2');

1 row created.

20:34:12 SQL>
20:34:12 SQL> pause Press ENTER to proceed to rollback so that the Unique Index
could be created
Press ENTER to proceed to rollback so that the Unique Index could be created
```

So session 45 did NOT have to wait on a Lock. It's insert was successful. (The table does not have a Unique Index so the same "key" value could be inserted).
Let's verify sessions 43 and 45 again.

```
20:34:24 SQL> pause Current Lock Holders on MY_TEST_TABLE
Current Lock Holders on MY_TEST_TABLE

20:34:24 SQL> @Lock_Holder MY_TEST_TABLE  HEMANT

no rows selected

20:34:24 SQL>
20:34:24 SQL> pause Current Active Transactions
Current Active Transactions

20:34:25 SQL> @active_transactions

       SID    SERIAL# SPID         USERNAME  PROGRAM          XIDUSN  USED_UBLK
USED_UREC
---------- ---------- ------------ --------- ------------ ---------- ----------
----------
LAST_CALL_ET
------------
SQL_TEXT
------------------------------------------------------------------------------
------------------------
        43         34 10420        HEMANT    sqlplus.exe           2          1
1
          45
insert into my_test_table values (1,'holder_s1')

        45         58 1104         HEMANT    sqlplus.exe           3          1
1
          15
insert into my_test_table values (1,'waiter_s2')


20:34:26 SQL>
20:34:26 SQL>
20:34:26 SQL> pause  Sessions currently Waiting on an Enqueue
```

```
Sessions currently Waiting on an Enqueue

20:34:27 SQL> select sid, event, seconds_in_wait from v$session_wait where event
like 'enq%' order by sid;

no rows selected

20:34:27 SQL>
20:34:27 SQL> pause Waiting and blocking session from v$session
Waiting and blocking session from v$session

20:34:28 SQL> select sid, blocking_session from v$session where username =
'HEMANT';

       SID BLOCKING_SESSION
---------- ----------------
        43
        45
        55

20:34:28 SQL>
```

See the listing above. Two active transactions. No apparently locks, no waiters or blockers.

Session 45 will do a rollback so that Sssion 43 will then be able to create a Unique Index.

```
20:34:35 SQL>
20:34:35 SQL> rollback;

Rollback complete.

20:34:35 SQL> pause Press ENTER to proceed
Press ENTER to proceed
```

Just check the transactions and locks before we proceed with the Unique Index.

```
20:34:42 SQL> pause Current Lock Holders on MY_TEST_TABLE
Current Lock Holders on MY_TEST_TABLE

20:34:42 SQL> @Lock_Holder MY_TEST_TABLE  HEMANT

no rows selected

20:34:42 SQL>
20:34:42 SQL> pause Current Active Transactions
Current Active Transactions

20:34:43 SQL> @active_transactions

       SID    SERIAL# SPID         USERNAME  PROGRAM          XIDUSN  USED_UBLK
USED_UREC
---------- ---------- ------------ --------- ------------ ---------- ----------
----------
LAST_CALL_ET
------------
SQL_TEXT
------------------------------------------------------------------------------
------------------------
        43         34 10420        HEMANT    sqlplus.exe           2          1
```

```
1
          60
insert into my_test_table values (1,'holder_s1')


20:34:43 SQL>
20:34:43 SQL>
20:34:43 SQL> pause  Sessions currently Waiting on an Enqueue
Sessions currently Waiting on an Enqueue

20:34:44 SQL> select sid, event, seconds_in_wait from v$session_wait where event
like 'enq%' order by sid;

no rows selected

20:34:44 SQL>
20:34:44 SQL> pause Waiting and blocking session from v$session
Waiting and blocking session from v$session

20:34:44 SQL> select sid, blocking_session from v$session where username =
'HEMANT';

       SID BLOCKING_SESSION
---------- ----------------
        43
        45
        55

20:34:44 SQL>
```

Session 43 is now committing the single row (remember, 45 has done a “rollback”) and then create a Unique Index. Next, it creates a new “Insert Transaction”.

```
20:34:48 SQL>
20:34:48 SQL> commit;

Commit complete.

20:34:48 SQL>
20:34:48 SQL> REM  Now let's see what locks would the presence of a unique index
cause
20:34:48 SQL>
20:34:48 SQL> create unique index my_test_table_uk1 on my_test_table (col1);

Index created.

20:34:48 SQL>
20:34:48 SQL> insert into my_test_table values (2,'holder_s1');

1 row created.

20:34:48 SQL>
20:34:48 SQL> pause Press ENTER to proceed
Press ENTER to proceed
```

Verify the state of Session 43’s transaction before we proceed to Session 45.

```
20:34:53 SQL>
20:34:53 SQL> pause Current Lock Holders on MY_TEST_TABLE
```

```
Current Lock Holders on MY_TEST_TABLE

20:34:54 SQL> @Lock_Holder MY_TEST_TABLE  HEMANT

no rows selected

20:34:54 SQL>
20:34:54 SQL> pause Current Active Transactions
Current Active Transactions

20:34:54 SQL> @active_transactions

       SID    SERIAL# SPID         USERNAME  PROGRAM          XIDUSN  USED_UBLK
USED_UREC
---------- ---------- ------------ --------- ------------ ---------- ----------
----------
LAST_CALL_ET
------------
SQL_TEXT
------------------------------------------------------------------------------
------------------------
        43         34 10420        HEMANT    sqlplus.exe           9          1
2
           5
insert into my_test_table values (2,'holder_s1')


20:34:54 SQL>
20:34:54 SQL>
20:34:54 SQL> pause  Sessions currently Waiting on an Enqueue
Sessions currently Waiting on an Enqueue

20:34:56 SQL> select sid, event, seconds_in_wait from v$session_wait where event
like 'enq%' order by sid;

no rows selected

20:34:56 SQL>
20:34:56 SQL> pause Waiting and blocking session from v$session
Waiting and blocking session from v$session

20:34:57 SQL> select sid, blocking_session from v$session where username =
'HEMANT';

       SID BLOCKING_SESSION
---------- ----------------
        43
        45
        55

20:34:57 SQL>
```

Here's the crux of the whole story. Session 45 attempts to insert a row. Under normal circumstances, (ie if Session 43 had actually done a COMMIT, our second transaction would immediately fail with an ORA-0001 error ("duplicate value" or "unique constraint violated"). However, we see now that Session 45 is "waiting" {potentially forever} because there is a "lock" held by Session 43 for the same Unique Key value (col1=2). This is where your user (Session 45) would complain that his transaction is taking a long time. If his insert

was based on a query you might suspect the performance of the query – and *that would take you on a wild goose chase* attempting to tune the query ! {Go ahead, build a test case where Session 45 is actually doing an "INSERT .. AS SELECT" but the INSERT is for COL1=2 and see it for yourself !}. Only if you look at waits (as we see later below, in BLUE) do you find out there is a Lock Wait. Huh ? A Lock Wait on an Insert you say. Inserts don't have to wait on Locks ! Well, this case demonstrates that it is possible to wait on a lock even when inserting what seem to be "new values". Guess what ? Even if you were to query the table from a 3rd of 4th session, you would not see the row with COL1=2 **because** Session 43 has not yet committed it's insert..

```
20:35:01 SQL>
20:35:01 SQL> REM  Now let's see what locks would the presence of a unique index
cause
20:35:01 SQL>
20:35:01 SQL> insert into my_test_table values (2,'waiter_s2');
insert into my_test_table values (2,'waiter_s2')
```

<seeming to be "hanging" here>

Having waited 7 seconds, I now check Locks , Transactions and Blockers.

```
20:35:08 SQL>
20:35:08 SQL> pause Current Lock Holders on MY_TEST_TABLE
Current Lock Holders on MY_TEST_TABLE

20:35:09 SQL> @Lock_Holder MY_TEST_TABLE  HEMANT

Object             Lock Type          Lock Held   Lock Req.         ID1       ID2
Oracle SID
-------------------- -------------------- ----------- ----------- ----------
---------- ----------
MY_TEST_TABLE       Transaction         Exclusive   None          655369      730
45
MY_TEST_TABLE       Transaction         None        Share         589856      723
45
MY_TEST_TABLE       DML                 Row-X (SX)  None           54969        0
45

20:35:09 SQL>
20:35:09 SQL> pause Current Active Transactions
Current Active Transactions

20:35:12 SQL> @active_transactions

       SID    SERIAL# SPID         USERNAME  PROGRAM          XIDUSN  USED_UBLK
USED_UREC
---------- ---------- ------------ --------- ------------ ---------- ----------
----------
LAST_CALL_ET
------------
SQL_TEXT
------------------------------------------------------------------------------
------------------------
        43         34 10420        HEMANT    sqlplus.exe           9          1
2
          23
insert into my_test_table values (2,'holder_s1')

        45         58 1104         HEMANT    sqlplus.exe          10          1
1
```

```
          12
insert into my_test_table values (2,'waiter_s2')


20:35:12 SQL>
20:35:12 SQL>
20:35:12 SQL> pause  Sessions currently Waiting on an Enqueue
Sessions currently Waiting on an Enqueue

20:35:14 SQL> select sid, event, seconds_in_wait from v$session_wait where event
like 'enq%' order by sid;

       SID EVENT                                                         SECONDS_IN_WAIT
---------- ----------------------------------------------------------------
---------------
        45 enq: TX - row lock contention                                              15

20:35:14 SQL>
20:35:14 SQL> pause Waiting and blocking session from v$session
Waiting and blocking session from v$session

20:35:15 SQL> select sid, blocking_session from v$session where username =
'HEMANT';

       SID BLOCKING_SESSION
---------- ----------------
        43
        45               43
        55

20:35:15 SQL>
```

Aah ! Now I see that Session 45's wait is an Enqueue Wait and that Session 43 is "blocking" Session 45's INSERT.

Let Session 43 commit so that session 45's "wait" ends and Session 45 attempts to proceed. (Are we in for another surprise ?).

```
20:35:22 SQL>
20:35:22 SQL> commit;

Commit complete.

20:35:22 SQL>
20:35:22 SQL> select * from my_test_table;

      COL1 COL2
---------- ------------
         1 holder_s1
         2 holder_s1

20:35:22 SQL>
```

Yes ! We have another surprise. It is NOW that Session 45 "fails" on the Unique Constraint -- because the COMMIT by Session 43 has made the row with COL1=2 quite visible !

```
ERROR at line 1:
ORA-00001: unique constraint (HEMANT.MY_TEST_TABLE_UK1) violated
```

```
20:35:22 SQL>
20:35:22 SQL> pause Press ENTER to proceed
Press ENTER to proceed
```

Just run our normal queries to lock for Locks, Transactions, Blockers and we find nothing now.
(Session 43 has committed and completed it's transaction, Session 45 has failed (on ORA-001) and rolled-back it's transaction).

```
20:35:35 SQL>
20:35:35 SQL> pause Current Lock Holders on MY_TEST_TABLE
Current Lock Holders on MY_TEST_TABLE

20:35:35 SQL> @Lock_Holder MY_TEST_TABLE  HEMANT

no rows selected

20:35:36 SQL>
20:35:36 SQL> pause Current Active Transactions
Current Active Transactions

20:35:36 SQL> @active_transactions

no rows selected

20:35:36 SQL>
20:35:36 SQL>
20:35:36 SQL> pause  Sessions currently Waiting on an Enqueue
Sessions currently Waiting on an Enqueue

20:35:36 SQL> select sid, event, seconds_in_wait from v$session_wait where event
like 'enq%' order by sid;

no rows selected

20:35:36 SQL>
20:35:36 SQL> pause Waiting and blocking session from v$session
Waiting and blocking session from v$session

20:35:36 SQL> select sid, blocking_session from v$session where username =
'HEMANT';

       SID BLOCKING_SESSION
---------- ----------------
        43
        45
        55

20:35:36 SQL>
```

Rollback in Session 45, just to be doubly safe (as "good practice" I always explicitly issue ROLLBACKs on my "test" transactions!)

```
20:35:39 SQL>
20:35:39 SQL> rollback;

Rollback complete.

20:35:39 SQL>
```