## ABOUT LOCKING & ISSUES

**MySQL Lock Modes**

MySQL implements standard row-level locking where there are two types of locks:

**1.A Shared (S) lock permits a transaction to read [ SELECT ] a row.**

**2.An Exclusive (X) lock permits a transaction to Update or Delete or Insert a row.**

## Read vs. Write locks

* Read locks are “shared” locks which prevent a write lock is being acquired but not other read locks.
* Write locks are “exclusive ” locks that prevent any other lock of any kind.

**LOCK and UNLOCK TABLES**

To lock a table, you specify its name after the LOCK TABLES keywords. In addition,

you specify the type of lock, either READ or WRITE.

mysql> LOCK TABLES emp READ;

Query OK, 0 rows affected (0.01 sec)

mysql> LOCK TABLES emp WRITE;

Query OK, 0 rows affected (0.00 sec)

To release a lock for a table, you use the following statement:

mysql>UNLOCK TABLES;

**Examples :-**

**Create database dbadb;**

**Use dbadb;**

create table hrtbl(id int,name char(30),city varchar(30));

insert into hrtbl values (1,'ramya','bangalore');

insert into hrtbl values (2,'sravya','hyderabad');

insert into hrtbl values (3,'raghu','Tirupati');

**create table payroll(id int,dept char(30),age int,salary varchar(20));**

insert into payroll values (1,'Hr Dept',25,'20,000');

insert into payroll values (2,'IT Dept',26,'30,000');

insert into payroll values (3,'SEO Dept',42,'25,000');

## Read Locks

A READ lock has the following features:

A READ lock for a table can be acquired by multiple sessions at the same time.

In addition,other sessions can read data from the table without acquiring the lock.

The session that holds the READ lock can only read data from the table, but cannot write.

Other sessions also cannot write data to the table until the READ lock is released.

The write operations from another session will be put into the waiting states until the READ lock is released.

If the session is terminated, either normally or abnormally, MySQL will release all the locks implicitly. This feature is also relevant for the WRITE lock.

Let’s have a look at how the READ lock works in the following examples

In the first session, first, connect to the dbadb database and use the CONNECTION\_ID()

function to get the current connection id as follows:

mysql> SELECT CONNECTION\_ID();

+-----------------+

| CONNECTION\_ID() |

+-----------------+

| 5 |

+-----------------+

Then, Insert a new row into the hrtbl table.

insert into hrtbl values (4,'Suresh','Bangalore');

Next, query the data the hrtbl table.

**SELECT \* FROM hrtbl;**

Now you use the LOCK TABLE statement.

LOCK TABLE hrtbl READ;

Finally, in the same session, if you try to insert a new row into the hrtbl table, you will get an error message.

Mean to say once READ lock is acquired, you cannot write data into the table within the same session also.

mysql> insert into dbadb.hrtbl values (4,'jaanu','Bangalore');

ERROR 1099 (HY000): Table 'hrtbl' was locked with a READ lock and can't be updated

mysql>

**mysql> SHOW OPEN TABLES IN dbadb;**

All open tables in the table cache are listed, but the IN\_USE column indicates of the table is locked.

The NAME\_LOCKED columns indicates whether the table name is locked,

due to a drop or rename request.

mysql> SHOW OPEN TABLES IN DBADB;

+----------+---------+--------+-------------+

| Database | Table | In\_use | Name\_locked |

+----------+---------+--------+-------------+

| dbadb | payroll | 0 | 0 |

| dbadb | hrtbl | 1 | 0 |

+----------+---------+--------+-------------+

2 rows in set (0.01 sec)

Let us check the READ lock from a different **Session 2**, connect to the dbadb and check the connection id:

mysql> SELECT CONNECTION\_ID();

+-----------------+

| CONNECTION\_ID() |

+-----------------+

| 11 |

+-----------------+

1 row in set (0.00 sec)

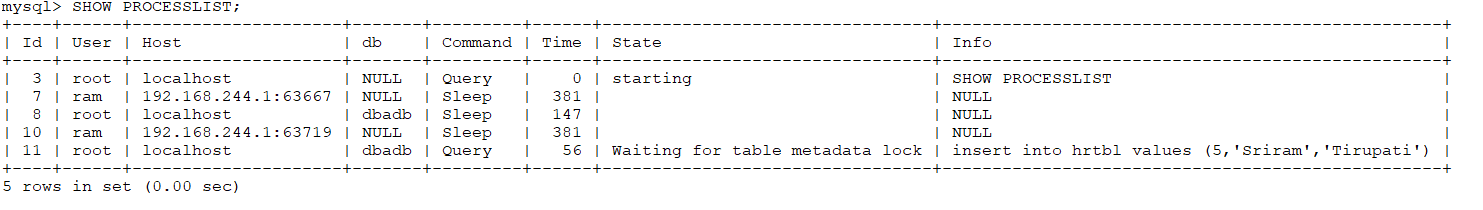
mysql> insert into hrtbl values (5,'Sriram','Tirupati');

waiting ....

The insert operation from the **session 2** is in the waiting state because a READ lock is already

acquired on the hrtbl table by the first session and it has not released yet.

You can see the information from the SHOW PROCESSLIST statement



Connection 11, is waiting , once the releases the id-5 will be insert.

Go back to the session 1 and release the lock by using the UNLOCK TABLES statement.

After you release the READ lock from the first session, the INSERT operation in the second session executed.

**Session 1**

mysql> unlock tables;

Query OK, 0 rows affected (0.00 sec)

mysql> select \* from hrtbl;

+------+--------+-----------+

| id | name | city |

+------+--------+-----------+

| 1 | ramya | bangalore |

| 2 | sravya | hyderabad |

| 3 | raghu | Tirupati |

| 4 | Suresh | Bangalore |

| 5 | Sriram | Tirupati |

+------+--------+-----------+

5 rows in set (0.00 sec)

**Write Locks**

A WRITE lock has the following features:

* The only session that holds the lock of a table can read and write data from the table.
* Other sessions cannot read data from and write data to the table until the WRITE lock is released.

Session:-

* First, put a WRITE lock from session 1.

LOCK TABLE hrtbl WRITE;

mysql> insert into hrtbl values (6,'mamatha','Hyderabad');

Query OK, 1 row affected (0.00 sec)

It works. Next, read data from the hrtbl table.

mysql> select \* from hrtbl;

+------+---------+-----------+

| id | name | city |

+------+---------+-----------+

| 1 | ramya | bangalore |

| 2 | sravya | hyderabad |

| 3 | raghu | Tirupati |

| 4 | Suresh | Bangalore |

| 5 | Sriram | Tirupati |

| 6 | mamatha | Hyderabad |

+------+---------+-----------+

6 rows in set (0.00 sec)

After that, from the session 2, try to write and read data:

mysql> insert into hrtbl values (7,'jaggu babu','Hyderabad');

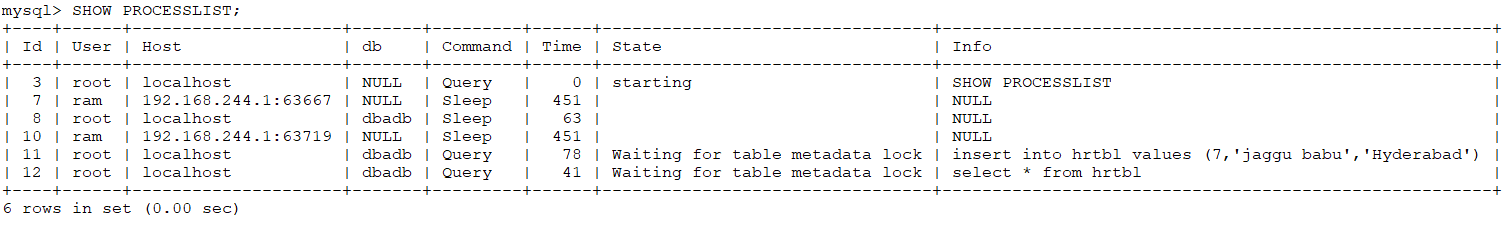
**waiting ...**

from the session 3 , try to select

**mysql> select \* from hrtbl;**

**waiting...**

from the SuperUser, You can check the SHOW PROCESSLIST statement.



Connection\_11 , Connection\_12 is waiting to run the queries

Finally, release the lock from the session 1

mysql> UNLOCK TABLES;

Query OK, 0 rows affected (0.00 sec)

you will see all pending operations from session 2 and Session 3 executed, illustrates the result:

one record inserted from session 2 and from session 3 he can view(select) the result

mysql> select \* from hrtbl;

+------+------------+-----------+

| id | name | city |

+------+------------+-----------+

| 1 | ramya | bangalore |

| 2 | sravya | hyderabad |

| 3 | raghu | Tirupati |

| 4 | Suresh | Bangalore |

| 5 | Sriram | Tirupati |

| 6 | mamatha | Hyderabad |

| 7 | jaggu babu | Hyderabad |

+------+------------+-----------+

7 rows in set (4 min 54.50 sec)

## How to kill, if any app\_user connection taking more time

## mysql> Select concat('KILL ',id,';') from information\_schema.processlist

## where user='app\_user';

## +------------------------+

## | concat('KILL ',id,';') |

## +------------------------+

## | KILL 11; |

## | KILL 12; |

## | KILL 7; |

## +------------------------+

## 3 rows in set (0.04 sec)

## mysql> KILL 8;

## Query OK, 0 rows affected (0.00 sec)

**Deadlock Example:-**

CREATE TABLE `student` ( `id` INT NOT NULL AUTO\_INCREMENT, `name` VARCHAR(255)

NOT NULL, `marks` INT NOT NULL, PRIMARY KEY (`id`) ) ENGINE=InnoDB;

INSERT INTO student (id, name, marks) VALUES (1, "jyothi", 50);

INSERT INTO student (id, name, marks) VALUES (2, "sravya", 80);

First Window , Update

BEGIN;

UPDATE student SET marks=marks-12 WHERE id =1; X [shared lock ] acquired on 1

Second Window Update Again

BEGIN;

UPDATE student SET marks=marks+5 WHERE id = 2; --X [shared lock ] acquired on 2

UPDATE student SET marks=marks+20 WHERE id = 1;-- LOCK WAIT!

COMMIT;

First Window (continued)

UPDATE student SET marks=marks+3 WHERE id=2; DEADLOCK!

COMMIT;

**If you want make it record in error log, Enable the variable**

**SET GLOBAL innodb\_print\_all\_deadlocks=1;**

Or **innodb\_print\_all\_deadlocks=1** add in my.cnf file and restart the server