Table Locking

MySQL database provides a multi-user environment, that allows multiple clients to access the database at the same time. To run this environment smoothly, MySQL introduced the concept of locks.

A client in a session can lock a certain table they are working on, in order to prevent other clients from using the same table. This process will avoid any data losses that might occur when multiple users work on the same table simultaneously.

A client can lock a table and unlock it whenever needed. However, if a table is already locked by a client session, it cannot be accessed by other client sessions until it is released.

Locking Tables in MySQL

You can restrict the access to records of the tables in MYSQL by locking them. These locks are used to keep other sessions away from modifying the tables in the current session.

MySQL sessions can acquire or release locks on the table only for itself. To lock a table using the MySQL LOCK TABLES Statement you need have the TABLE LOCK and SELECT privileges.

These locks are used to solve the concurrency problems. There are two kinds of MYSQL table locks –

- **READ LOCK** – If you apply this lock on a table the write operations on it are restricted. i.e., only the sessions that holds the lock can write into this table.
- **WRITE LOCK** – This lock allows restricts the sessions (that does not possess the lock) from performing the read and write operations on a table.
- Syntax
- Following is the syntax of the MySQL LOCK TABLES Statement –

```
LOCK TABLES table_name [READ | WRITE];
```

Unlocking Tables in MySQL

Once the client session is done using/accessing a MySQL table, they must unlock the table for other client sessions to use it. To do so, you can use the MySQL UNLOCK TABLE statement. This will release the table until other sessions lock it again.

Syntax

Following is the syntax of the MySQL UNLOCK TABLES Statement –

```
UNLOCK TABLES;
```

Example

Let us start with creating a table named **CUSTOMERS** that contains the details as shown below –

```
CREATE TABLE CUSTOMERS (
   ID INT AUTO_INCREMENT,
   NAME VARCHAR(20) NOT NULL,
   AGE INT NOT NULL,
   ADDRESS CHAR (25),
   SALARY DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

```
INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES
 (1, 'Ramesh', 32, 'Ahmedabad', 2000.00 ),
 (2, 'Khilan', 25, 'Delhi', 1500.00 );
```

Create another table named BUYERS using the following query –

```
CREATE TABLE BUYERS (
   B_ID INT AUTO_INCREMENT,
   B_NAME VARCHAR(20) NOT NULL,
   B_AGE INT NOT NULL,
   B_ADDRESS CHAR (25),
   B_SALARY DECIMAL (18, 2),
   PRIMARY KEY (B_ID)
);
```

Following queries inserts records into the BUYERS table using the INSERT INTO SELECT statement. Here, we are trying to insert records from the CUSTOMERS table to BUYERS table.

Locking and Unlocking:

Here before the transfer, we are acquiring the write lock on the BUYERS table to which we are inserting records and acquiring read lock on the CUSTOMERS table from which we are inserting records. Finally, after the transfer we are releasing the records.

```
LOCK TABLES CUSTOMERS READ, BUYERS WRITE;


INSERT INTO BUYERS (B_ID, B_NAME, B_AGE, B_ADDRESS, B_SALARY)
  SELECT
    ID, NAME, AGE, ADDRESS, SALARY
  FROM
    CUSTOMERS
  WHERE
    ID = 1 AND NAME = 'Ramesh';
INSERT INTO BUYERS (B_ID, B_NAME, B_AGE, B_ADDRESS, B_SALARY)
  SELECT
    ID, NAME, AGE, ADDRESS, SALARY
  FROM
    CUSTOMERS
  WHERE
    ID = 2 AND NAME = 'Khilan';

UNLOCK TABLES;
```

```
SELECT * FROM BUYERS;
```

Table Locking Using a Client Program

Besides locking a table in a MySQL database with a MySQL query, we can also use a client program to perform the LOCK TABLES operation.

Syntax

Following are the syntaxes to Lock a table in MySQL in various programming languages –

```
sql="LOCK TABLES table_name [READ | WRITE]";
cursorObj.execute(sql);
```

```
import mysql.connector
#establishing the connection
connection = mysql.connector.connect(
   host='localhost',
   user='root',
   password='password',
   database='tut'
)
table_name = 'tutorials_tbl'
#Creating a cursor object
cursorObj = connection.cursor()
lock_table_query = f"LOCK TABLES {table_name} WRITE"
cursorObj.execute(lock_table_query)
print(f"Table '{table_name}' is locked successfully.")
cursorObj.close()
connection.close()
```