Experiment No.10
Implementation and demonstration of Transaction and Concurrency control techniques using locks
Date of Performance:
Date of Submission:

Aim :- Write a query to lock and unlock a table for transaction and concurrency control. **Objective :-** To learn locking of tables for transaction processing and concurrency control. **Theory:**

A lock is a mechanism associated with a table used to restrict the unauthorized access of the data in a table. MySQL allows a client session to acquire a table lock explicitly to cooperate with other sessions to access the table's data. MySQL also allows table locking to prevent unauthorized modification into the same table during a specific period.

Table Locking in MySQL is mainly used to solve concurrency problems. It will be used while running a transaction, i.e., first read a value from a table (database) and then write it into the table (database).

MySQL provides two types of locks onto the table, which are:

READ LOCK: This lock allows a user to only read the data from a table.

WRITE LOCK: This lock allows a user to do both reading and

writing into a table. The following is the syntax that allows us to

acquire a table lock explicitly:

LOCK TABLES table_name [READ | WRITE];

The following is the syntax that allows us to release a lock for a table

in MySQL: UNLOCK TABLES;

Conclusion: Locking and unlocking of tables is achieved and verified using insert command in the same table of a database system.

1. Explain Transaction and Concurrency control techniques using locks.

   Ans Transactions are units of work in a database, ensuring Atomicity (all-or-nothing), Consistency (data integrity), Isolation (transactions appear serial), and Durability (committed changes are permanent). They group operations like fund transfers or inventory updates.

   Concurrency control techniques manage simultaneous transaction execution:

   - Locking: Transactions acquire shared (read) or exclusive (write) locks on data objects.

   - Two-Phase Locking (2PL): Transactions acquire locks in two phases: growing (acquiring locks) and shrinking (releasing locks).

   - Deadlock Detection and Resolution: Mechanisms to identify and resolve deadlocks where transactions wait for each other.

   - Timestamp-based Concurrency Control: Assigns timestamps to transactions,

determining execution order.

- Optimistic Concurrency Control: Transactions execute without locks, resolving conflicts before committing changes.

These techniques ensure safe, efficient transaction execution in multi-user databases, maintaining data integrity and scalability.