**Batch: A1 Roll No.: 1911004**

**Experiment / assignment / tutorial No. 10**

**Grade: AA / AB / BB / BC / CC / CD /DD**

|  |
| --- |
| **Title:**  **Implementation of Concurrency Control Protocols** |

**Objective:** To understand Transaction, Transaction Control Protocols and its implementation. Implement Lock based protocol.

**Expected Outcome of Experiment:**

CO 5: Formulate and demonstrate the transaction, concurrency and recovery techniques

**Books/ Journals/ Websites referred:**

1. Dr. P.S. Deshpande, SQL and PL/SQL for Oracle 10g.Black book, Dreamtech Press

2. www.db-book.com

3. Korth, Slberchatz, Sudarshan : “Database Systems Concept”, 5th Edition , McGraw Hill

4. Elmasri and Navathe,”Fundamentals of database Systems”, 4th Edition,PEARSON Education.

5. <https://dev.mysql.com/doc/refman/8.0/en/innodb-transaction-isolation-levels.html>

**Resources used:**

**Theory**

In a multiprogramming environment where multiple transactions can be executed simultaneously, it is highly important to control the concurrency of transactions.

**Concurrency control is provided in a database to:**

1. enforce isolation among transactions.
2. preserve database consistency through consistency preserving execution of transactions.
3. resolve read-write and write-read conflicts.

**Concurrency control protocols can be broadly divided into two categories −**

1. Lock based protocols
2. Time stamp based protocols

**Lock-based Protocols**

Database systems equipped with lock-based protocols use a mechanism by which any transaction cannot read or write data until it acquires an appropriate lock on it. Locks are of two kinds −

1. Binary Locks − A lock on a data item can be in two states; it is either locked or unlocked.
2. Shared/exclusive − This type of locking mechanism differentiates the locks based on their uses. If a lock is acquired on a data item to perform a write operation, it is an exclusive lock. Allowing more than one transaction to write on the same data item would lead the database into an inconsistent state. Read locks are shared because no data value is being changed.

**Timestamp Ordering Protocol**

The timestamp-ordering protocol ensures serializability among transactions in their conflicting read and write operations. This is the responsibility of the protocol system that the conflicting pair of tasks should be executed according to the timestamp values of the transactions.

The timestamp of transaction Ti is denoted as TS(Ti).

Read time-stamp of data-item X is denoted by R-timestamp(X).

Write time-stamp of data-item X is denoted by W-timestamp(X).

**Timestamp ordering protocol works as follows −**

If a transaction Ti issues a read(X) operation −

If TS(Ti) < W-timestamp(X)

Operation rejected.

If TS(Ti) >= W-timestamp(X)

Operation executed.

All data-item timestamps updated.

If a transaction Ti issues a write(X) operation −

If TS(Ti) < R-timestamp(X)

Operation rejected.

If TS(Ti) < W-timestamp(X)

Operation rejected and Ti rolled back.

Otherwise, operation executed.

**Implementation of Lock Protocol:**

We will be implementing the Lock Protocol on the table User which has been defined as follows:

create table if not exists User(

user\_name varchar(20) primary key ,

name varchar(20) not null,

phone\_number bigint not null,

Email\_id varchar(30) not null,

Dob date not null,

password varchar(20) not null,

age int not null,

gender char

);

This is the normal way to insert values which will be executed since there is no Lock on the table as of now:

insert into User(

user\_name,

name,

phone\_number,

Email\_id,

Dob,

password,

age,

gender

)

values(

" PQR "," PQR ","5241452586"," pqr@g.com ","1970-10-11",

" pqr@123 ","50","M"

);

# Now we issue a ‘READ’ lock on the table which will not allow any connection to write the table

# 

# After acquiring a READ lock we try to insert a value into the table

# 

# Unlocking the table

# 

# Post Lab Questions:

# Explain pitfalls of 2PL (Two Phase Locking) Protocol

# 1. Two-phase locking may also limit the amount of concurrency that occur in a schedule because a Transaction may not be able to release an item after it has used it.

# 2. This may be because of the protocols and other restrictions we may put on the schedule to ensure serializability, deadlock freedom and other factors.

# 3. This is the price we have to pay to ensure serializability and other factors, hence it can be considered as a bargain between concurrency and maintaining the ACID properties.

# Conclusion (In your own words):

# In this program we were successfully able to understand the Lock based protocols and implement it on our database using different connections.