

# **DBMS - Experiment 10**

Name: **Ayush Jain**

Sapid: **60004200132**

Div./Batch: **B/B1**

Branch: **Computer Engineering**

## **AIM:**

Implement Transaction and Concurrency control techniques.

## **Theory:**

A transaction is a logical unit of processing in a DBMS which entails one or more database access operation. In a nutshell, database transactions represent real-world events of any enterprise.

All types of database access operation which are held between the beginning and end transaction statements are considered as a single logical transaction. During the transaction the database is inconsistent. Only once the database is committed the state is changed from one consistent state to another.

Concurrency control is the procedure in DBMS for managing simultaneous operations without conflicting with each another. Concurrent access is quite easy if all users are just reading data. There is no way they can interfere with one another. Though for any practical database, would have a mix of reading and WRITE operations and hence the concurrency is a challenge.

Concurrency control is used to address such conflicts which mostly occur with a multi-user system. It helps you to make sure that database transactions are performed concurrently without violating the data integrity of respective databases.

Therefore, concurrency control is a most important element for the proper functioning of a system where two or multiple database transactions that require access to the same data, are executed simultaneously.

## Concurrency Control Protocols

Different concurrency control protocols offer different benefits between the amount of concurrency they allow and the amount of overhead that they impose.

- Lock-Based Protocols
- Two Phase
- Timestamp-Based Protocols
- Validation-Based Protocols

## Concurrency control using Mysql Locks screenshots:

### 1. User 1 acquires read lock

#### a. User 1 reads from table song

The screenshot shows the MySQL Workbench interface. The query window contains the following SQL statements:

```
1 use music_app;
2 lock tables song read, artist read;
3 select * from song;
```

The result grid displays the following data:

song_id	song_title	length	album_id	genre	artist_id
5	Peaches	00:03:18	1	Pop	3
6	Love nwantiti	00:03:08	2	Pop	9
7	Hello	00:04:55	2	soul	9
8	When we were young	00:04:50	2	soul	9
16	Smile	00:03:16	3	Hip-hop	7
18	Galway Girl	00:02:50	4	Pop	4
19	Perfect	00:04:23	4	Pop	4
21	Industry Baby	00:03:32	10	Pop rap	10
49	Halle luja	00:02:40	10	Pop	10
50	Kacha Badam	00:00:45	10	Pop	10

The output pane shows the following execution steps:

#	Time	Action	Message	Duration / Fetch
4	23:43:28	use music_app	0 row(s) affected	0.000 sec
5	23:43:28	lock tables song read, artist read	0 row(s) affected	0.000 sec
6	23:43:28	select * from song LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

#### b. User 1 writes in the song table

MySQL Workbench

User 1 x User 2 x

File Edit View Query Database Server Tools Scripting Help

Navigator

Filter objects

exam

just\_crypto

music\_app

album

artist

genre

playlist

playlist\_content

podcast

song

subscription

user

Views

Stored Procedures

Functions

sakila

sys

world

Administration Schemas

Information

No object selected

Object Info Session

Query 1

```

1 use music_app;
2 lock tables song read, artist read;
3 select * from song;
4 insert into song (song_id, song_title, length) values(67, "Gaadi meri 2 seater...", "0:1:59");

```

Result Grid

song_id	song_title	length	album_id	genre	artist_id
5	Peaches	00:03:18	1	Pop	3
6	Love nwantiti	00:03:08	1	Pop	9
7	Hello	00:04:55	2	soul	9
8	When we were young	00:04:50	2	soul	9
16	Smile	00:03:16	3	Hip-hop	7
18	Galway Girl	00:02:50	4	Pop	4
19	Perfect	00:04:23	4	Pop	4
21	Industry Baby	00:03:32	10	Pop rap	10
49	Halle luja	00:02:40	10	Pop	10
50	Kache Badam	00:00:45	10	Pop	10

Output

Action Output

#	Time	Action	Message	Duration / Fetch
8	23:45:16	lock tables song read, artist read	0 row(s) affected	0.000 sec
9	23:45:16	select * from song LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
10	23:45:16	insert into song (song_id, song_title, length) values(67, "Gaadi meri 2 seater...", "0:1:59")	Error Code: 1100. Table 'album' was not locked with LOCK TABLES	0.000 sec

Query interrupted

c. User 2 reads from the table song

MySQL Workbench

User 1 x User 2 x

File Edit View Query Database Server Tools Scripting Help

Navigator

Filter objects

exam

just\_crypto

music\_app

album

artist

genre

playlist

playlist\_content

podcast

song

subscription

user

Views

Stored Procedures

Functions

sakila

sys

world

Administration Schemas

Information

Schema: exam

Object Info Session

Query 1

```

1 use music_app;
2 select * from song;

```

Result Grid

song_id	song_title	length	album_id	genre	artist_id
5	Peaches	00:03:18	1	Pop	3
6	Love nwantiti	00:03:08	1	Pop	9
7	Hello	00:04:55	2	soul	9
8	When we were young	00:04:50	2	soul	9
16	Smile	00:03:16	3	Hip-hop	7
18	Galway Girl	00:02:50	4	Pop	4
19	Perfect	00:04:23	4	Pop	4
21	Industry Baby	00:03:32	10	Pop rap	10
49	Halle luja	00:02:40	10	Pop	10
50	Kache Badam	00:00:45	10	Pop	10

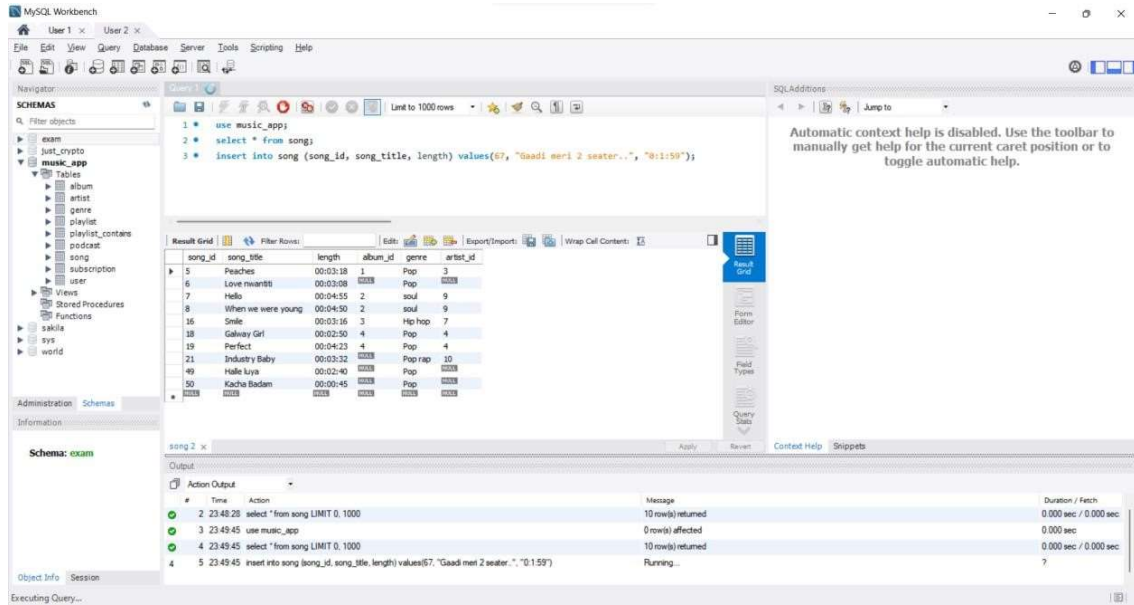
Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	23:48:28	use music_app	0 row(s) affected	0.000 sec
2	23:48:28	select * from song LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

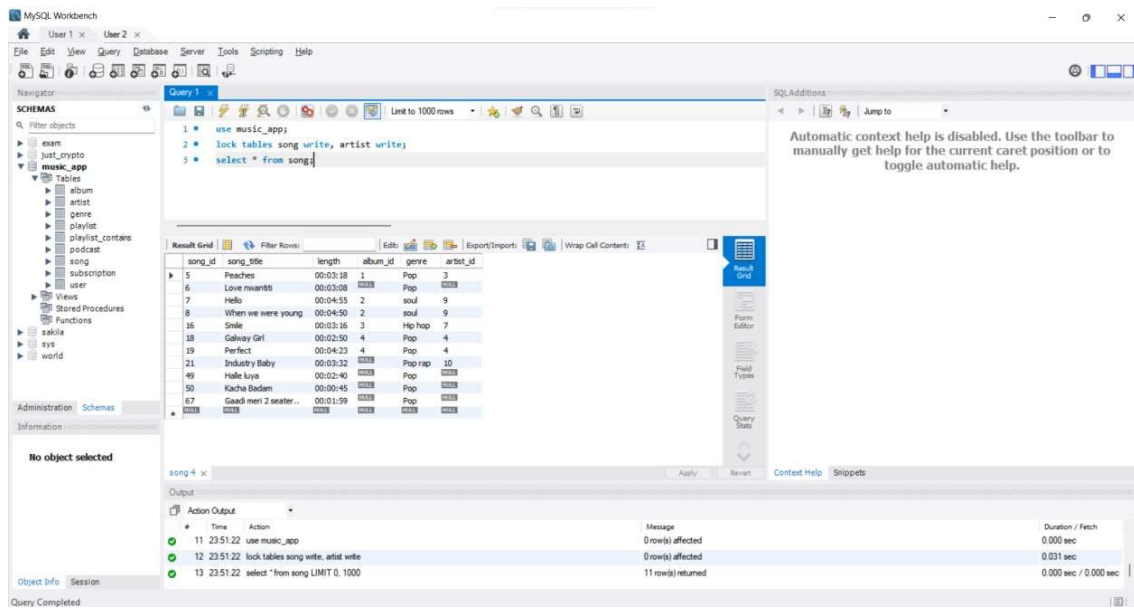
Query Completed

d. User 2 writes in table song



## 2. User 1 acquires write lock

### a. User 1 reads from table song



### b. User 1 writes in table song

MySQL Workbench

User 1 x User 2 x

File Edit View Query Database Server Tools Scripting Help

Navigator

Filter objects

exam

just\_crypto

music\_app

album

artist

genre

playlist

playlist\_contans

podcast

subscription

user

Views

Stored Procedures

Functions

sakila

sys

world

Administration Schemas

Information

No object selected

Object Info Session

Query 1

Limit to 1000 rows

```

1 use music_app;
2 lock tables song write, artist write;
3 select * from song;
4 insert into song (song_id, song_title, length) values(69, "espideeman espideeman", "0:2:20");

```

Result Grid

song_id	song_title	length	album_id	genre	artist_id
5	Peaches	00:03:18	1	Pop	3
6	Love meant	00:03:08	2	Pop	9
7	hello	00:04:15	2	soul	9
8	When we were young	00:04:50	2	soul	9
16	Smile	00:03:16	3	Hip hop	7
18	Galway Girl	00:02:50	4	Pop	4
19	Perfect	00:04:23	4	Pop	4
21	Industry Baby	00:03:32	4	Pop rap	10
49	Halle luya	00:02:40	4	Pop	4
50	Kacha Badam	00:00:45	4	Pop	4
67	Gaadi meri 2 seater..	00:01:59	4	Pop	4

song 8 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
17	23:54:50	insert into song (song_id, song_title, length) values(67, "Gaadi meri 2 seater..", "0:1:59")	Error Code: 1062. Duplicate entry '67' for key 'song.PRIMARY'	0.000 sec
18	23:55:44	use music_app	0 row(s) affected	0.000 sec
19	23:55:44	lock tables song write, artist write	0 row(s) affected	0.000 sec
20	23:55:44	select * from song LIMIT 0, 1000	11 row(s) returned	0.000 sec / 0.000 sec
21	23:55:44	insert into song (song_id, song_title, length) values(69, "espideeman espideeman", "0:2:20")	1 row(s) affected	0.000 sec

Query Completed

c. User 2 reads in table song

MySQL Workbench

User 1 x User 2 x

File Edit View Query Database Server Tools Scripting Help

Navigator

Filter objects

exam

just\_crypto

music\_app

album

artist

genre

playlist

playlist\_contans

podcast

song

subscription

user

Views

Stored Procedures

Functions

sakila

sys

world

Administration Schemas

Information

Schema: exam

Object Info Session

Query 1

Limit to 1000 rows

```

1 use music_app;
2 select * from song;

```

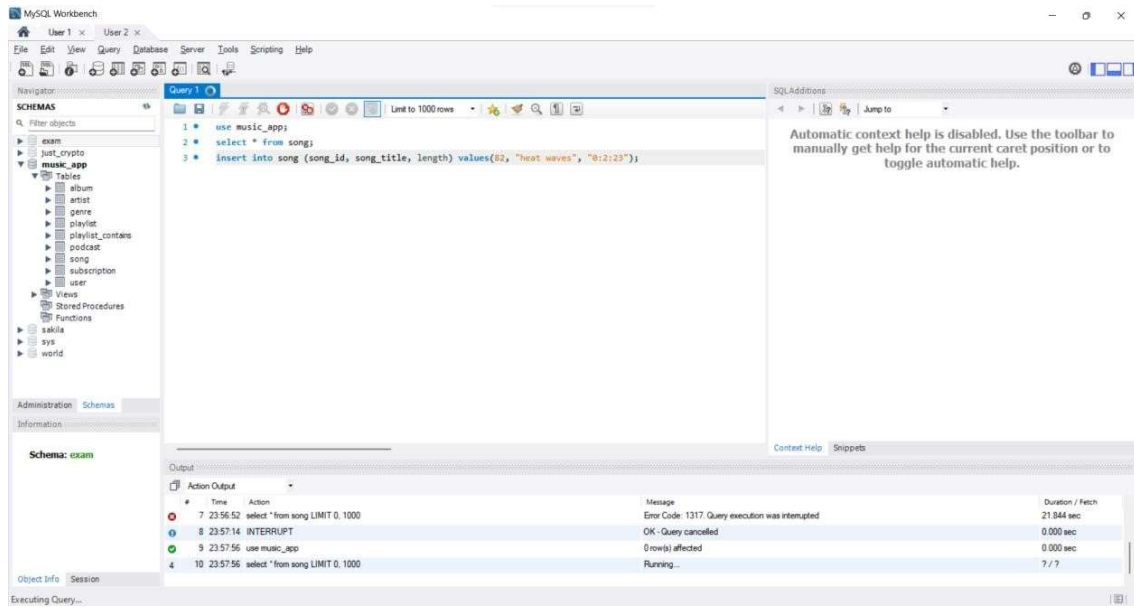
Output

Action Output

#	Time	Action	Message	Duration / Fetch
4	23:49:45	select * from song LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
5	23:49:45	insert into song (song_id, song_title, length) values(67, "Gaadi meri 2 seater..", "0:1:59")	Error Code: 1013. Lost connection to MySQL server during query	30.031 sec
6	23:56:52	use music_app	0 row(s) affected	0.000 sec
7	23:56:52	select * from song LIMIT 0, 1000	Running...	7 / 7

Executing Query...

d. User 2 writes in table song



## Conclusion:

Locks of mysql is used to demonstrate concurrency.