Experiment No.7
Perform DCL and TCL commands
Date of Performance:
Date of Submission:

Aim :- Write a query to implement Data Control Language(DCL) and Transaction Control Language(TCL) commands

Objective :- To learn DCL commands like Grant and Revoke privileges to the user and TCL commands to commit the transactions and recover it using rollback and save points.

Theory:

Data Control Language:

DCL commands are used to grant and take back authority from any database user.

- Grant
- Revoke

a. Grant: It is used to give user access privileges to a database.

Example

```
1. GRANT SELECT, UPDATE ON MY_TABLE TO
   SOME_USER, ANOTHER_USER;
```

b. Revoke: It is used to take back permissions from the user.

Example

```
1. REVOKE SELECT, UPDATE ON MY_TABLE FROM USER1, USER2;
```

Transaction Control Language

TCL commands can only use with DML commands like INSERT, DELETE and UPDATE only.

These operations are automatically committed in the database that's why they cannot be used while creating tables or dropping them.

Here are some commands that come under TCL:

- COMMIT
- ROLLBACK
- SAVEPOINT

a. Commit: Commit command is used to save all the transactions to the database. Syntax:

```
1. COMMIT;
```

Example:

```
1. DELETE FROM CUSTOMERS
2. WHERE AGE = 25;
3. COMMIT;
```

b. Rollback: Rollback command is used to undo transactions that have not already been saved to the database.

Syntax:

```
1. ROLLBACK;
```

Example:

```
1. DELETE FROM CUSTOMERS
2. WHERE AGE = 25;
3. ROLLBACK;
```

c. SAVEPOINT: It is used to roll the transaction back to a certain point without rolling back the entire transaction.

Syntax:

```
2. SAVEPOINT SAVEPOINT_NAME;
```

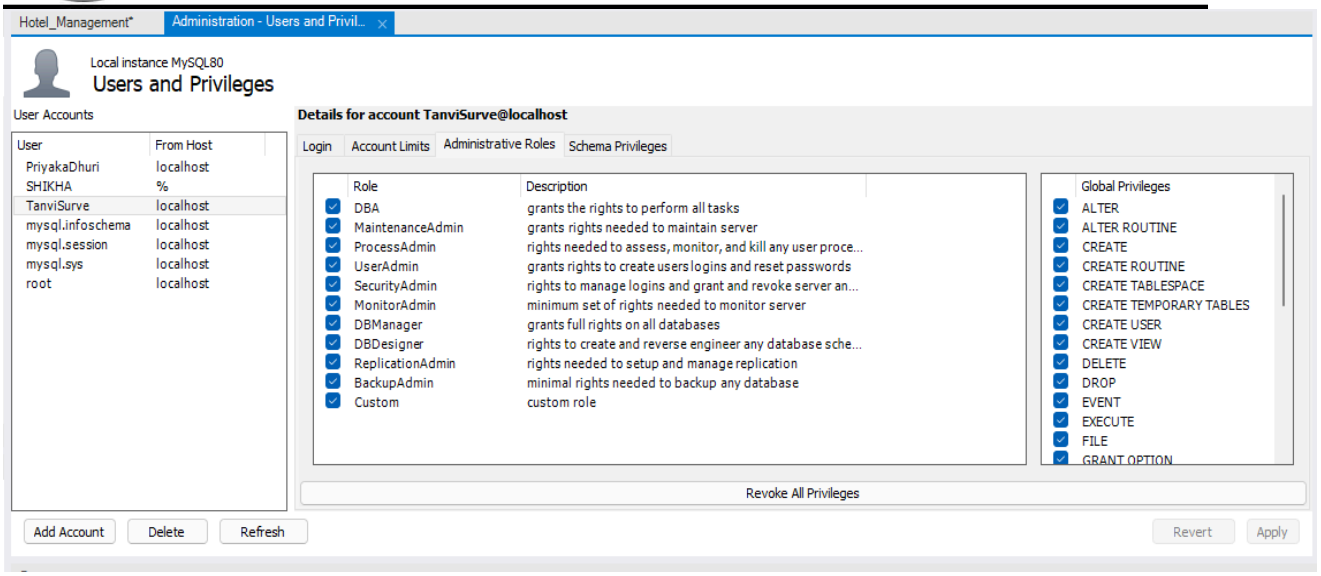
Implementation:

DCL:-

GRANT :-

```
CREATE USER
'TanviSurve'@'localhost' IDENTIFIED BY 'Tanvi',
'PriyakaDhuri'@'localhost' IDENTIFIED BY 'Priyanka';

GRANT ALL ON *.* TO 'TanviSurve'@'localhost'WITH GRANT OPTION;
```



TCL:-

COMMIT:-

```
1  Create database Hotel_Management;
2  use Hotel_Management;
3  Select * from customer;
4  COMMIT;
5
```

ROLLBACK:-

```
1  Create database Hotel_Management;
2  use Hotel_Management;
3  Select * from customer;
4  ROLLBACK;
```

SAVEPOINT:-

```
1  Create database Hotel_Management;
2  use Hotel_Management;
3  Select * from customer;
4  SAVEPOINT Tanvi;
5
```

Conclusion:

1. **Explain about issues faced during rollback in mysql and how it got resolved.**

In MySQL, rollback operations can encounter several issues, primarily related to the management of transactions and the handling of locks. Here are some common issues and their resolutions:

Deadlocks:

a) Issue: Deadlocks can occur when two or more transactions are waiting for each other to release locks on resources, resulting in a deadlock situation where none of the transactions can proceed.

b) Resolution: MySQL's InnoDB engine automatically detects and resolves deadlocks by rolling back the transaction that is less likely to be committed. Additionally, application-level deadlock detection and retry mechanisms can be implemented to handle deadlocks gracefully.

c) Lock Wait Timeout:

d) Issue: If a transaction is waiting for a lock on a resource for too long, it may encounter a lock wait timeout error, indicating that the transaction has been waiting for a lock for longer than the specified timeout period.

e) Resolution: Adjust the innodb_lock_wait_timeout parameter in MySQL configuration to increase the timeout period if necessary. Alternatively, optimize queries and transactions to reduce lock contention and minimize the likelihood of lock wait timeouts.

Resource Limit Exceeded:

a) Issue: Rollback operations may encounter resource limits, such as maximum undo log size or transaction size limits, especially for large transactions or in high-load environments.

b) Resolution: Increase the relevant resource limits in MySQL configuration, such as innodb_undo_log_truncate or innodb_undo_tablespaces, to accommodate larger transactions. Additionally, consider breaking down large transactions into smaller units to reduce resource usage.

Transaction Aborted:

a) Issue: Transactions may be aborted due to various reasons, such as deadlock resolution, lock wait timeouts, or resource limits being exceeded.

b) Resolution: Implement error handling and retry mechanisms in application code to handle transaction aborts gracefully. Retry the transaction or provide appropriate feedback to users, depending on the specific scenario.

2. **Explain how to create a user in sql.**

To create a user in SQL, you typically use the CREATE USER statement. However, the exact syntax and options available may vary depending on the specific database management system (DBMS) you are using. Here's a general overview of how to create a user in SQL:

Syntax:
In most SQL databases, the syntax for creating a user is similar to the following:

```
CREATE USER username IDENTIFIED BY 'password';
```

Parameters:
username: Specifies the name of the user you want to create.
password: Specifies the password for the user account. Note that in some databases, password-related options may vary or additional security measures like hashing may be required.
Example (using MySQL syntax):

```
CREATE USER 'newuser'@'localhost' IDENTIFIED BY 'password';
```

This example creates a new user named 'newuser' who can connect from the 'localhost' host, identified by the password 'password'.

Privileges:
After creating a user, you may need to grant specific privileges to the user to allow them to perform certain actions, such as accessing databases or executing queries. Privileges can be granted using the GRANT statement.

Additional Options:
Depending on the DBMS you are using, there may be additional options available when creating a user, such as specifying resource limits, setting default roles, or defining authentication mechanisms.
It's important to note that creating users and managing user permissions typically requires administrative privileges in the database. Additionally, it's essential to follow best practices for user management, such as using strong passwords and granting only the necessary privileges to users based on their roles and responsibilities.