



**Vidyavardhini's College of Engineering and Technology**

**Department of Artificial Intelligence & Data Science**

---

<b>Experiment No.7</b>
Perform DCL and TCL commands
Date of Performance:
Date of Submission:



# Vidyavardhini's College of Engineering and Technology

## Department of Artificial Intelligence & Data Science

---

**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



# Vidyavardhini's College of Engineering and Technology

## Department of Artificial Intelligence & Data Science

---

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:

Data Control Language:

```
155 • CREATE TABLE admin (  
156     admin_id INTEGER PRIMARY KEY,  
157     bank_id INTEGER  
158 );  
159  
160 -- Grant privileges to a user  
161 • GRANT SELECT, INSERT, UPDATE, DELETE ON admin TO 'root'@'localhost';  
162  
163 -- Revoke privileges from a user  
164 • REVOKE SELECT, INSERT, UPDATE, DELETE ON admin FROM 'root'@'localhost';  
165
```

### Transaction Control Language:

```
166      -- Start a transaction
167 •   START TRANSACTION;
168
169      -- Make some changes (e.g., update cash_available in the ATM table)
170 •   UPDATE atm SET cash_available = 52000.00 WHERE location = 'ATM 1';
171
172      -- Set a savepoint
173 •   SAVEPOINT after_cash_update;
174
175      -- Commit the transaction
176 •   COMMIT;
177
178      -- Rollback changes made in the transaction
179 •   START TRANSACTION;
180
181      -- Set another savepoint
182 •   SAVEPOINT before_withdraw;
183
184      -- Simulate a withdrawal operation
185 •   DELETE FROM transactions WHERE atm_id = 1 AND transaction_type = 'withdrawal';
186
187      -- Rollback to the savepoint before the withdrawal operation
188 •   ROLLBACK TO SAVEPOINT before_withdraw;
```

### Conclusion:

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

In MySQL, rollback refers to the process of undoing changes made to the database during a transaction that has been aborted or rolled back. When a rollback is initiated, MySQL needs to revert the database to its state before the transaction began. There can be various issues that arise during rollback, and MySQL employs several mechanisms to handle them:

**Lock Contention:** Rollback operations may face issues due to lock contention, where other transactions hold locks on data being rolled back, potentially causing delays or deadlocks.

**Resource Limitations:** Insufficient memory or disk space can hinder rollback operations, leading to failures or stalling of the process.

**Performance Overhead:** Rollback operations can impose a significant performance overhead, especially in large databases or high-concurrency environments, due to increased disk I/O and CPU usage.

**Deadlock Situations:** Deadlocks may occur during rollback, where transactions are blocked indefinitely due to conflicting lock requests.

**Strategies to Address Issues:**

**Deadlock Detection:** MySQL employs mechanisms to detect and resolve deadlock situations.

**Resource Optimization:** Configuration parameters allow adjustment of memory and disk space usage for rollback operations, optimizing resource utilization.

**Performance Enhancements:** Techniques like write-ahead logging minimize disk I/O during rollback, enhancing efficiency.

**Data Integrity Measures:** Transaction logs and recovery mechanisms ensure data integrity during rollback, allowing recovery from failures or errors encountered during the process.

**Administrator Actions:** Administrators can fine-tune MySQL configurations to optimize rollback performance and ensure reliable data management in the face of rollback-related challenges. Adjustments may include tuning deadlock detection settings, optimizing resource allocation, and monitoring performance metrics to identify bottlenecks.

2. Explain how to create a user in sql.

To create a user in SQL, we use the CREATE USER statement

Syntax:

CREATE USER username [IDENTIFIED BY 'password'];

Explanation:

**CREATE USER:** This keyword is used to create a new user in the database.

**username:** Specifies the name of the user you want to create.

**IDENTIFIED BY 'password' (optional):** Specifies the password for the user. This part is optional, and some database systems might have different mechanisms for managing user authentication.

Example (for MySQL):

CREATE USER 'john' IDENTIFIED BY 'password123';

This example creates a user named 'john' with the password 'password123' in MySQL.

**Privileges:**

After creating the user, you may want to grant specific privileges to the user, such as SELECT, INSERT, UPDATE, or DELETE on certain tables. You can grant privileges using the GRANT statement. For example, to grant SELECT privileges on a table named employees to the user 'john', you can use:

GRANT SELECT ON employees TO 'john';

Additionally, you can use the WITH GRANT OPTION clause to allow the user to grant these privileges to other users.

