Transaction Control Commands

In MySQL, transaction control commands allow you to manage transactions within your database. Transactions ensure that a series of SQL statements are executed as a single unit of work, either all successfully completed (committed) or all undone (rolled back) in case of failure. Here are the primary transaction control commands in MySQL:

1. START TRANSACTION, COMMIT, and ROLLBACK

START TRANSACTION

- **Purpose**: Begins a new transaction explicitly.
- **Syntax**:

  ```
  START TRANSACTION;
  ```

- **Notes**:
  - In MySQL, transactions are implicitly started with the first executable SQL statement. However, you can use START TRANSACTION to explicitly begin a transaction if needed.

COMMIT

- **Purpose**: Saves the changes made during the current transaction permanently.
- **Syntax**:

  ```
  COMMIT;
  ```

- **Notes**:
  - COMMIT makes all changes made during the current transaction visible and permanent.
  - After committing, you cannot roll back the changes.

ROLLBACK

- **Purpose**: Undoes the changes made during the current transaction.
- **Syntax**:

  ```
  ROLLBACK;
  ```

- **Notes**:
  - ROLLBACK reverses all modifications made during the current transaction.
  - It cancels the transaction and restores the state of the database as it was before the transaction began.

2. SAVEPOINT and ROLLBACK TO SAVEPOINT

SAVEPOINT

- **Purpose**: Sets a named transaction savepoint with a specific name within the current transaction.
- **Syntax**:

  ```
  SAVEPOINT savepoint_name;
  ```

- **Notes**:
  - SAVEPOINT allows you to create points within the current transaction that you can later roll back to if needed.
  - Useful for partial rollback within transactions.

ROLLBACK TO SAVEPOINT

- **Purpose**: Rolls back the current transaction to a named savepoint.
- **Syntax**:

  ```
  ROLLBACK TO SAVEPOINT savepoint_name;
  ```

- **Notes**:
  - ROLLBACK TO SAVEPOINT undoes all changes made after the specified savepoint within the current transaction.

- This command allows for more granular control over rollback operations.

Example Usage

Here's an example demonstrating the use of transaction control commands in MySQL:

```
CREATE TABLE Employees (
    EmployeeID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Position VARCHAR(50),
    Salary DECIMAL(10, 2)
);

==========================================

-- Start a new transaction explicitly
START TRANSACTION;

-- Insert some records
INSERT INTO Employees9 (FirstName, LastName, Position, Salary)
VALUES ('Venkat', 'S', 'Manager', 60000),
       ('Amit', 'K', 'Developer', 50000);

select * from Employees;

-- Save a named savepoint
SAVEPOINT before_update;

-- Update a record
UPDATE Employees
SET Salary = 55000
WHERE FirstName = 'Amit' AND LastName = 'K';

select * from Employees;
```

```
-- Rollback to the savepoint if necessary
ROLLBACK TO SAVEPOINT before_update;

select * from Employees;



-- Delete a record
SAVEPOINT before_delete;

DELETE FROM Employees
WHERE FirstName = 'Amit' AND LastName = 'K';

select * from Employees;

ROLLBACK TO SAVEPOINT before_delete;

select * from Employees;

-- Commit the transaction
COMMIT;
```

In this example:

- We start a transaction with START TRANSACTION.
- Perform some operations (INSERT, UPDATE, DELETE).
- Use SAVEPOINT to create a point (before_update) within the transaction.
- Roll back to before_update if needed using ROLLBACK TO SAVEPOINT.
- Finally, commit all changes with COMMIT.

Transaction control commands are essential for ensuring data integrity and consistency in database operations, especially when multiple SQL statements need to be treated as a single atomic unit of work.