SQL Commands Overview

SQL Commands:

SQL (Structured Query Language) is a language used to interact with databases. It allows you to create, modify, and manage data in a database. SQL commands can be categorized into five main types:

Types of Commands

SQL commands can be divided into the following types:

1. **Data Definition Language (DDL)**
2. **Data Manipulation Language (DML)**
3. **Data Query Language (DQL)**
4. **Transaction Control Language (TCL)**
5. **Data Control Language (DCL)**

Now, let's break down the commands for each category:

1.1 DDL- Data Definition Language (Building and Changing Database Structures)

DDL commands are used to define and manage the structure of database objects, such as tables, schemas, and indexes. These commands help in creating, altering, and deleting database structures.

Common DDL Commands:

CREATE: Creates a new database object (table, index, or schema).	**Example**: Suppose you're a part of an e-commerce company and need to store customer data. You can create a table to store customer information. **CREATE TABLE customers (** **customer_id INT PRIMARY KEY,** **name VARCHAR(100),** **email VARCHAR(100),** **created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP** **);** This command creates a new customers table with columns for customer ID, name, email, and the date/time the record was created.

ALTER: Modifies an existing database object (such as adding a new column or changing a data type).	**Example**: Later, your company wants to store the phone number for each customer. You can add a phone_number column to the customers table. **ALTER TABLE customers ADD phone_number VARCHAR(15);** This adds a new phone_number column to the customers table to store phone numbers.
DROP: Deletes an existing database object (like a table or a view).	**Example**: Suppose your company no longer needs a temporary table that was used for marketing purposes. You can drop it from the database. **DROP TABLE temp_customer_data;** This command deletes the temp_customer_data table permanently.
TRUNCATE: Removes all rows from a table but keeps the table structure intact.	**Example**: At the end of the month, you want to clear out the old transaction data for reporting purposes, but you want to keep the transactions table structure for future use. **TRUNCATE TABLE transactions;** This command removes all records from the transactions table, but the table itself is still available for future use.

1.2 DML- Data Manipulation Language (Managing Data in Tables)

DML commands allow you to manipulate the data stored in the database. These commands help you add, update, delete, or merge data.

Common DML Commands:

INSERT: Adds new records (rows) to a table.	**Example**: When a new customer signs up on your e-commerce site, their information needs to be added to the customers table. **INSERT INTO customers (customer_id, name, email, phone_number)** **VALUES (1, 'janardhan kcs', 'janardhan.jk@example.com', '123-456-7890');** This command adds a new customer record with their details to the customers table.
UPDATE: Modifies existing records in a table.	**Example**: After Alice Johnson's phone number changes, you can update her record in the customers table. **UPDATE customers** **SET phone_number = '987-654-3210'** **WHERE customer_id = 1;** This command updates the phone number of the customer with customer_id = 1 to a new value.
DELETE: Removes one or more records from a table.	**Example**: If a customer decides to delete their account, you can remove their record from the customers table. **DELETE FROM customers** **WHERE customer_id = 1;** This command deletes the customer with customer_id = 1 from the customers table.
MERGE: Combines INSERT, UPDATE, and DELETE in a single operation based on a condition.	**Example**: Your company might be merging data from two systems, where some customers already exist and some are new. You can use

	MERGE to update existing customers and insert new ones. **MERGE INTO customers c** **USING new_customers n** **ON (c.customer_id = n.customer_id)** **WHEN MATCHED THEN** **UPDATE SET c.name = n.name, c.email = n.email** **WHEN NOT MATCHED THEN** **INSERT (customer_id, name, email, phone_number)** **VALUES (n.customer_id, n.name, n.email, n.phone_number);** This command updates existing customer records and inserts new ones from the new_customers table.

1.3 DQL- Data Query (Retrieval) Language (Fetching Data from the Database)

DRL commands are used to retrieve data from the database. They allow you to view or analyze the data stored in tables.

Common DRL Command:

SELECT: Retrieves data from one or more tables.	**Example**: To view the list of all customers, you can run a simple SELECT query. **SELECT * FROM customers;** This command retrieves all columns and rows from the customers table.
	Example with Condition: If you want to see the details of customers who have made a purchase above $100, you can use a condition. **SELECT name, email** **FROM customers c**

	JOIN orders o ON c.customer_id = o.customer_id **WHERE o.total_amount > 100;** This command retrieves the names and emails of customers who have made large purchases, using a JOIN to combine the customers and orders tables.

1.4 TCL- Transaction Control Language (Managing Database Transactions)

TCL commands manage changes made by DML commands. They control the transaction process, ensuring data integrity.

- **Common TCL Commands:**

COMMIT: Saves all changes made in the current transaction	**Real-Time Example**: After processing a batch of customer updates or adding a set of new records, you can commit the changes to make them permanent. **COMMIT;** This saves all the changes made in the current transaction permanently in the database.
ROLLBACK: Reverts changes made since the last COMMIT.	**Real-Time Example**: If you make a mistake while updating customer records (e.g., updating the wrong field), you can rollback to undo the changes. **ROLLBACK;** This undoes all changes made in the current transaction since the last commit.
SAVEPOINT: Marks a specific point within a transaction to which you can later roll back.	**Real-Time Example**: You can set a savepoint before updating customer details. If anything goes wrong later, you can roll back to the savepoint.

	SAVEPOINT customer_update_savepoint; This sets a savepoint named customer_update_savepoint, which can be rolled back to if necessary.

1.5 DCL- Data Control Language (Managing Access Rights and Permissions)

DCL commands are used to control access to the data in the database. These commands help in granting or revoking permissions to users.

- **Common DCL Commands:**

GRANT: Gives specific permissions to a user or role.	**Real-Time Example**: You can grant a user permission to view customer data but not modify it. GRANT SELECT ON customers TO user1; This command grants the user1 permission to SELECT (view) data from the customers table.
REVOKE: Removes specific permissions from a user or role.	**Real-Time Example**: If a user no longer needs access to sensitive customer data, you can revoke their permission. REVOKE SELECT ON customers FROM user1; This command revokes the SELECT privilege on the customers table from user1

Summary

- **DDL (Data Definition Language)**: Used for creating, altering, and deleting database structures (e.g., tables).
- **DML (Data Manipulation Language)**: Used for inserting, updating, deleting, and merging data in tables.
- **DRL (Data Retrieval Language)**: Used for querying and retrieving data from the database.
- **TCL (Transaction Control Language)**: Manages transactions, ensuring changes are permanent or rolled back.
- **DCL (Data Control Language)**: Manages user permissions and access to database objects.

These SQL commands are essential for performing all types of operations in a database, from designing structures to managing data and controlling user access.