SQL COMMANDS

LEARNINGS

- SQL Commands
- Types of SQL Commands
  - DDL
  - DML
  - DCL
  - DTL
  - DQL

WHAT ARE SQL COMMANDS?

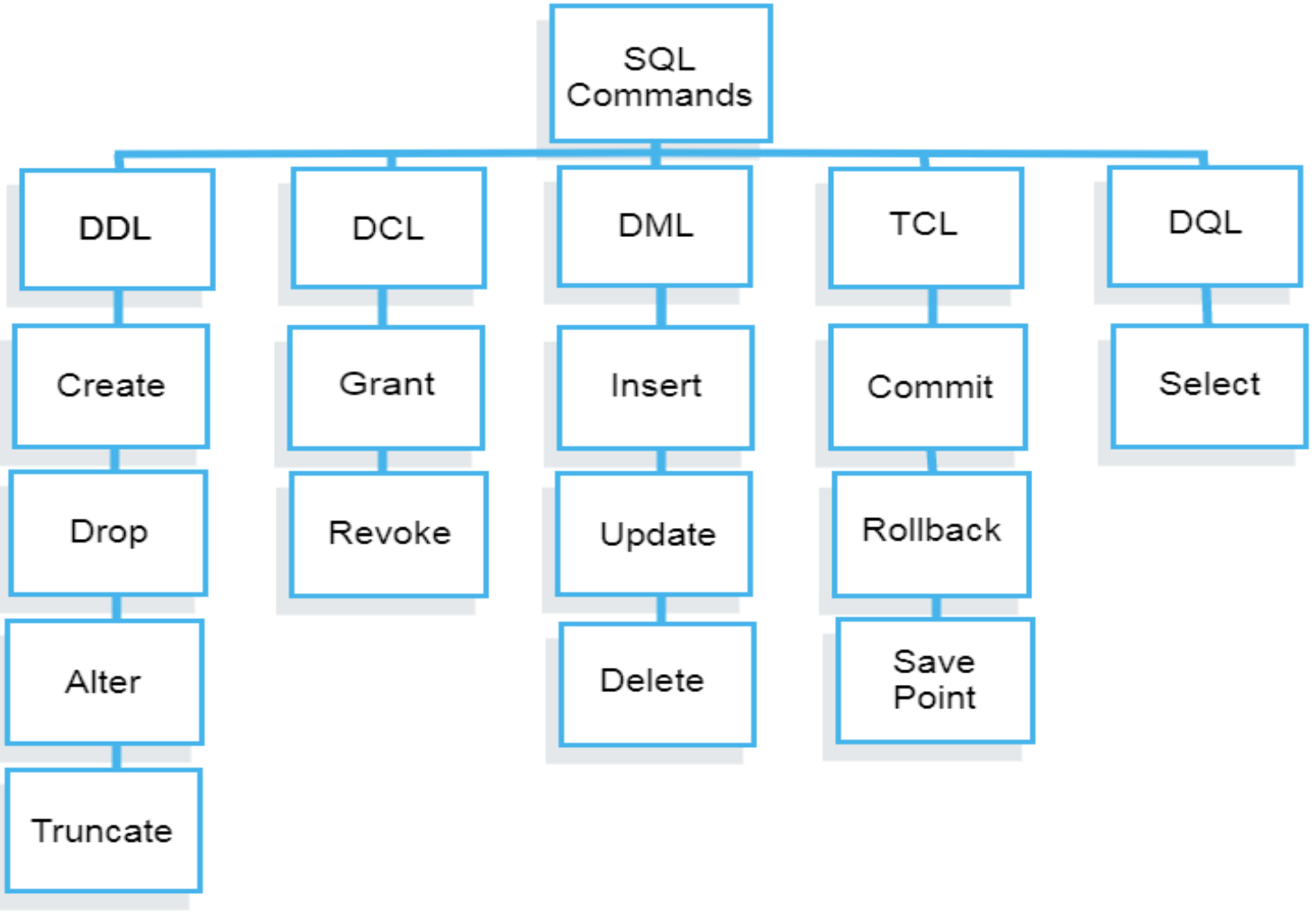
- SQL commands are the instructions used to communicate with a database to perform tasks, functions, and queries with data.
- SQL commands can be used to search the database and to do other functions like creating tables, adding data to tables, modifying data, and dropping tables.

TYPES OF SQL COMMANDS…

SQL Commands are divided into **five broad categories** –

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

TYPES OF SQL COMMANDS…



DATA DEFINITION LANGUAGE(DDL)

- DDL or Data Definition Language actually consists of SQL commands that can be used to **define and change** the database structure or schema.
- It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.
- All DDL commands are **auto-committed**. That means it saves all the changes **permanently** in the database.
- Five types of DDL commands in SQL are:
  - Create
  - Drop
  - Alter
  - Truncate
  - Rename

DDL COMMANDS

Five types of DDL commands in SQL are:

S.No	Command	Description
1	**CREATE**	to create the database or its objects (like table, index, function, views, store procedure, and triggers)
2	**ALTER**	to alter the structure of the database
3	**TRUNCATE**	to remove all records from a table, including all spaces allocated for the records are removed
4	**DROP**	used to delete objects from the database(hence delete both structure and records)
5	**RENAME**	to rename an object existing in the database

CREATE COMMAND...

CREATE Database:

- To create a database in RDBMS, **create** command is used.

Syntax:	Example:
CREATE DATABASE database_name ;	**CREATE DATABASE** iotaclass ;

After creating database you have to select database.

Syntax :
USE database_name;

CREATE COMMAND...

- Creating ***student_table*** with columns named *student_id, student_name* and *student_age.*
- We need to **mention datatype** of all the columns during Create Table command itself

Syntax :

```
CREATE TABLE table_name
(
  column_name1  DATATYPE NOT NULL,
  column_name2  DATATYPE,
  column_name3 DATATYPE
);
```

Example :

```
CREATE TABLE student_table
(
student_id INT NOT NULL,
student_name VARCHAR(255),
student_age INT
);
```

DATA TYPES

List of some of the most commonly used data types in SQL.

DATA TYPE	USED for Columns which will
INT	store **integer** values.
FLOAT	store **float** values.
VARCHAR(size)	A **VARIABLE length string** (can contain letters, numbers, and special characters). The size parameter specifies the maximum string length in characters
CHAR(size)	A **FIXED length stri**ng (can contain letters, numbers, and special characters).
DATE	store **date** values.
TEXT	store text which is generally lengthy (up to 2GB size)

ALTER COMMAND

- Alter command is used for altering the table structure, such as:
  - to **add** a column to existing table
  - to **rename** any existing column
  - to **change data type** of any column or to **modify** its size.
  - to **drop** a column from the table.

ALTER COMMAND...

Alter Command: ADD a new column:

Using ALTER Command we can **add a column** to any existing table. We can even add new column with constraints like Foreign Key in the existing table.

Syntax :	Example :
ALTER TABLE table_name **ADD** column_name ***datatype***;	**ALTER TABLE** student_table **ADD** city VARCHAR(255);

ALTER COMMAND...

Alter Command: ADD multiple new columns

- Using Alter Command we can even add multiple new columns to any existing table.

Syntax :	Example :
ALTER TABLE table_name **ADD**(column_name1 datatype, column_name2 datatype);	**ALTER TABLE** student_table **ADD**(father_name VARCHAR(60), mother_name VARCHAR(60));

ALTER COMMAND...

Alter Command: Add a new column after specified location

Using ALTER Command we can add a column to any existing table.

Syntax :	Example :
ALTER TABLE table_name **ADD** column column_name *datatype* **after** column_name **;**	**ALTER TABLE** student_table **ADD** column full_name VARCHAR(200) **after** father_name**;**

ALTER COMMAND...

Alter Command: ADD Column with DEFAULT value

- ALTER Command can add a new column to an existing table with a default values too. The default value is used when no value is inserted in the column.

Syntax :	Example :
ALTER TABLE table_name **ADD** column-name1 datatype1 **DEFAULT** default_value ;	**ALTER TABLE** student_table **ADD** city VARCHAR(255) **DEFAULT** 'INDORE';

ALTER COMMAND…

MODIFY an existing column:

- Alter command can also be used to modify data type of any existing column.

Syntax :	Example :
ALTER TABLE table_name **MODIFY** COLUMN column_name data_type;	**ALTER TABLE** student_table **MODIFY** COLUMN address VARCHAR(300);

ALTER COMMAND...

RENAME a Table:

- Using Alter command we can Rename an existing table.

Syntax :	Example :
RENAME TABLE old_table **TO** new_table;	**RENAME TABLE** student_table **TO** iota;

ALTER COMMAND...

Rename a column:

- Using Alter command we can Rename an existing column.

Syntax :	Example :
ALTER TABLE table_name **RENAME COLUMN** old_column_name **TO** new_col_name;	**ALTER TABLE** student_table **RENAME COLUMN** student_id **TO** roll_number**;**

ALTER COMMAND...

Rename a column:

- Using Alter command, **CHANGE** to **rename** and **change datatype** of an existing column in a table.

Syntax :	Example :
ALTER TABLE table_name **CHANGE** old_column_name new_column_name **datatype;**	**ALTER TABLE** student_table **CHANGE** student_id roll_number **INT;**

ALTER COMMAND...

DROP a Column:

- Alter Command can also be used to drop or remove column.

Syntax :	Example :
ALTER TABLE table_name **DROP COLUMN** column_name;	**ALTER TABLE** student_table **DROP COLUMN** city;

ALTER COMMAND...

DROP a Column with Constraint/key:

- If there is any constraint like a **Primary key/Foreign key constraint** applied on any column, which isn't allowing to drop a column then first we have to drop the constraint.
- As we can have more than one foreign key in a table, we require foreign key identifier to drop a particular foreign key.

Syntax :

```
ALTER TABLE table_name
DROP constraint_name identifier;
```

ALTER COMMAND...

DROP a Column with Constraint/key:

- **Example:** Creating tables with Primary and Foreign Keys

Table 1:

```
CREATE TABLE customer_table
(
  customer_id INT PRIMARY KEY,
  customer_name VARCHAR(255),
  city VARCHAR(255),
  state VARCHAR(255)
);
```

Table 2 :

```
CREATE TABLE order_table
(
  order_id INT PRIMARY KEY,
  order_number INT,
  customer_id INT,
  FOREIGN KEY(customer_id) REFERENCES
customer_table(customer_id)
);
```

ALTER COMMAND...

DROP a Column with Constraint/key:

- **Example:**

EXAMPLE 1:

```
ALTER TABLE order_table
DROP FOREIGN KEY `order_table_ibfk_1` ;
```

EXAMPLE 2:

```
ALTER TABLE order_table
DROP PRIMARY KEY;
```

TRUNCATE COMMAND

- Truncate Command **removes all the records** from a table, but this command **will not destroy the table's structure**.
- Logically, the TRUNCATE TABLE statement is like a **DELETE** statement without a WHERE clause that deletes all rows from a table, or a **sequence of DROP TABLE and CREATE TABLE statements**.
- TRUNCATE TABLE statement is **more efficient than the DELETE statement** because it drops and recreates the table instead of deleting rows one by one.
- If there is any FOREIGN KEY constraints from other tables which reference the table that you truncate, the TRUNCATE TABLE statement will fail.

Syntax :	Example :
TRUNCATE TABLE table_name;	**TRUNCATE TABLE** Iota_student;

DROP COMMAND

Drop database:

- It can be used to delete the complete database.

Syntax :	Example :
DROP DATABASE database_name;	**DROP DATABASE** iotaclass;

DROP COMMAND…

Drop table:

- Drop Command **completely removes a table** from the database.
- This command will also **destroy** the table structure and the data stored in it.

Syntax :	Example :
DROP TABLE table_name;	**DROP TABLE** student_table;

COMMENT STYLES IN MYSQL

MySQL Server supports three comment styles:

- From a **#** character at the start of a sentence, you want to comment.
- In MySQL, the **-- (double-dash)** comment style requires the second dash to be followed by at least one whitespace.
- From a **/* sequence to the following */ sequence**, as in the C programming language. This syntax enables a comment to extend over **multiple lines**.
- The following example demonstrates all three comment styles:

Example :

```
SELECT * FROM table;    # single-line comment
SELECT * FROM table;    -- This is a single-line comment
SELECT * FROM table;    /* this is a multiline comment */
```

THANK YOU