# Full Stack
# Web Programming

## SQL Commands: Part 1

Lesson 7


# Lesson outline

- Introduction
- DDL – Data Definition Language
- DQl – Data Query Language
- DML – Data Manipulation Language
- DCL – Data Control Language

# Introduction

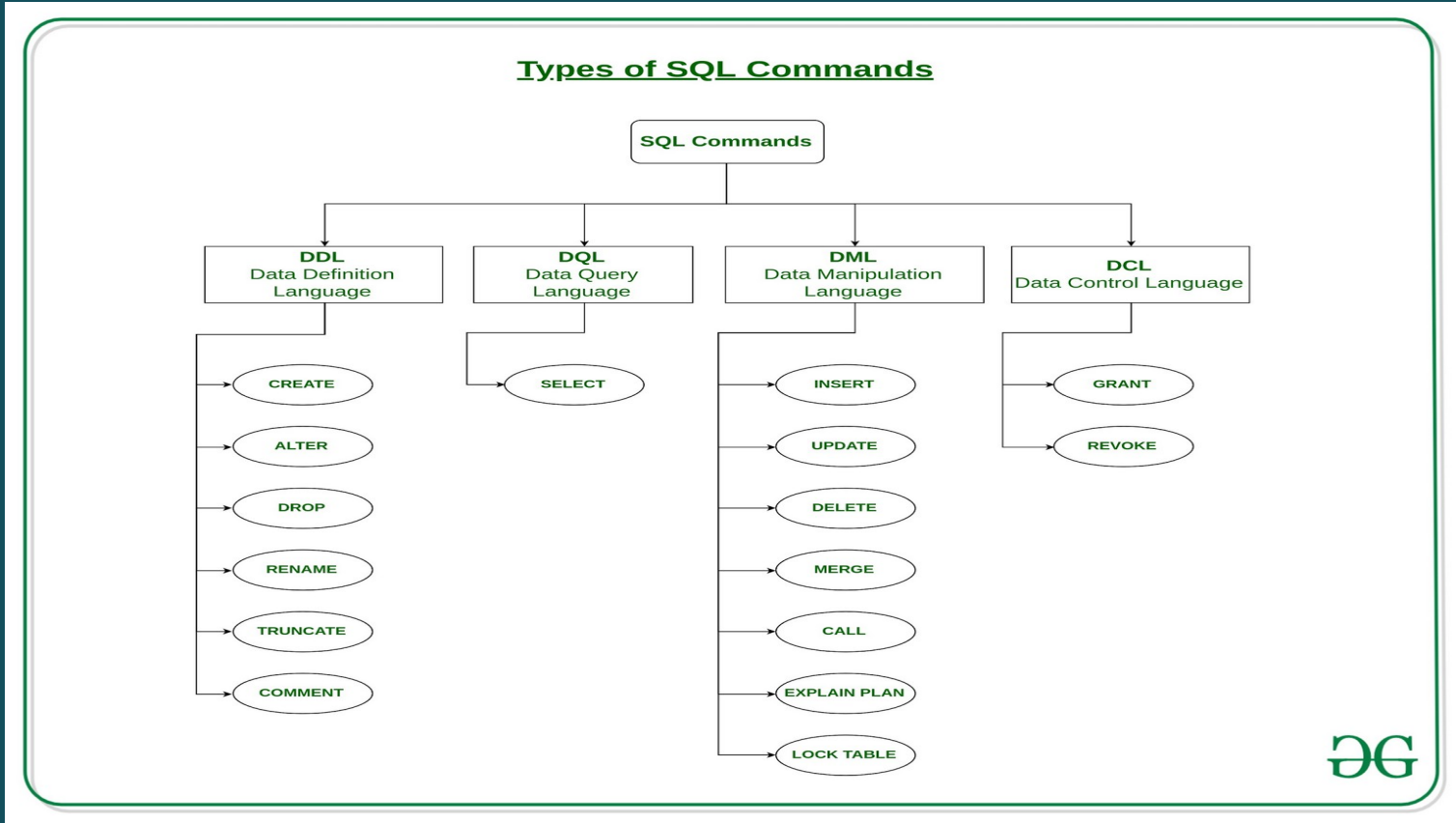
- Structured Query Language(SQL) as we all know is the database language by the use of which we can perform certain operations on the existing database and also we can use this language to create a database. SQL uses certain commands like Create, Drop, Insert etc. to carry out the required tasks.

These SQL commands are mainly categorized into four categories as:

- DDL – Data Definition Language
- DQl – Data Query Language
- DML – Data Manipulation Language
- DCL – Data Control Language

Though many resources claim there to be another category of SQL clauses TCL – Transaction Control Language. So we will see in detail about TCL as well.

# Introduction



# Data Definition Language (DDL)

DDL(Data Definition Language) : DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database 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.

# Data Definition Language (DDL)

Examples of DDL commands:

**CREATE** – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).

**DROP** – is used to delete objects from the database.

**ALTER** - is used to alter the structure of the database.

**TRUNCATE** – is used to remove all records from a table, including all spaces allocated for the records are removed.

**COMMENT** – is used to add comments to the data dictionary.

RENAME - is used to rename an object existing in the database

# DDL - SQL: CREATE

There are two CREATE statements available in SQL:
CREATE DATABASE
CREATE TABLE

CREATE DATABASE
A Database is defined as a structured set of data. So, in SQL the very first step to store the data in a well structured manner is to create a database. The CREATE DATABASE statement is used to create a new database in SQL.
Syntax:

```
CREATE DATABASE database_name;

database_name: name of the database.
```

# DDL - SQL: CREATE

Example Query:

This query will create a new database in SQL and name the database as my_database.

```
CREATE DATABASE my_database;
```

# DDL - SQL: CREATE

## CREATE TABLE

We have learned above about creating databases. Now to store the data we need a table to do that. The CREATE TABLE statement is used to create a table in SQL. We know that a table comprises of rows and columns. So while creating tables we have to provide all the information to SQL about the names of the columns, type of data to be stored in columns, size of the data etc. Let us now dive into details on how to use CREATE TABLE statement to create tables in SQL.

# DDL - SQL: CREATE

```
CREATE TABLE table_name
(
column1 data_type(size),
column2 data_type(size),
column3 data_type(size),
....
);
```

**table_name**:  name of the table.
**column1** name of the first column.
**data_type**: Type of data we want to store in the particular column.
For example,**int** for integer data.
**size**: Size of the data we can store in a particular column. For example if for a column we specify the data_type as int and size as 10 then this column can store an integer number of maximum 10 digits.

# DDL - SQL: CREATE

Example Query:
This query will create a table named Students with three columns, ROLL_NO, NAME and SUBJECT.

```
CREATE TABLE Students
(
ROLL_NO int(3),
NAME varchar(20),
SUBJECT varchar(20),
);
```

This query will create a table named Students. The ROLL_NO field is of type int and can store an integer number of size 3. The next two columns NAME and SUBJECT are of type varchar and can store characters and the size 20 specifies that these two fields can hold maximum of 20 characters

# DDL - SQL: DROP, TRUNCATE

DROP

DROP is used to delete a whole database or just a table.The DROP statement destroys the objects like an existing database, table, index, or view.
A DROP statement in SQL removes a component from a relational database management system (RDBMS).
Syntax:

```
DROP object object_name

Examples:
DROP TABLE table_name;
table_name: Name of the table to be deleted.

DROP DATABASE database_name;
database_name: Name of the database to be deleted.
```

# DDL - SQL: DROP, TRUNCATE

TRUNCATE

TRUNCATE statement is a Data Definition Language (DDL) operation that is used to mark the extents of a table for deallocation (empty for reuse). The result of this operation quickly removes all data from a table, typically bypassing a number of integrity enforcing mechanisms. It was officially introduced in the SQL:2008 standard.
The TRUNCATE TABLE mytable statement is logically (though not physically) equivalent to the DELETE FROM mytable statement (without a WHERE clause).
Syntax:

```
TRUNCATE TABLE  table_name;
table_name: Name of the table to be truncated.
DATABASE name - student_data
```

# DDL - SQL: DROP, TRUNCATE

DROP vs TRUNCATE

Truncate is normally ultra-fast and its ideal for deleting data from a temporary table.
Truncate preserves the structure of the table for future use, unlike drop table where the table is deleted with its full structure.
Table or Database deletion using DROP statement cannot be rolled back, so it must be used wisely.

# DDL - SQL: DROP, TRUNCATE

**Student**

| ROLL_NO | NAME | ADDRESS | PHONE | Age |
|---|---|---|---|---|
| 1 | Ram | Delhi | xxxxxxxxxx | 18 |
| 2 | RAMESH | GURGAON | xxxxxxxxxx | 18 |
| 3 | SUJIT | ROHTAK | xxxxxxxxxx | 20 |
| 4 | SURESH | Delhi | xxxxxxxxxx | 18 |
| 3 | SUJIT | ROHTAK | xxxxxxxxxx | 20 |
| 2 | RAMESH | GURGAON | xxxxxxxxxx | 18 |

# DDL - SQL: DROP, TRUNCATE

**Student_Details**

| ROLL_NO | Branch | Grade |
|---|---|---|
| 1 | Information Technology | O |
| 2 | Computer Science | E |
| 3 | Computer Science | O |
| 4 | Mechanical Engineering | A |

# DDL - SQL: DROP, TRUNCATE

To delete the whole database

```
DROP DATABASE student_data;
```

After running the above query whole database will be deleted.
To truncate Student_details table from student_data database.

```
TRUNCATE TABLE Student_details;
```

After running the above query Student_details table will be truncated, i.e, the data will be deleted but the structure will remain in the memory for further operations.

# DDL - SQL: ALTER (ADD, DROP, MODIFY)

ALTER TABLE is used to add, delete/drop or modify columns in the existing table. It is also used to add and drop various constraints on the existing table.

ALTER TABLE – ADD

ADD is used to add columns into the existing table. Sometimes we may require to add additional information, in that case we do not require to create the whole database again, ADD comes to our rescue.

Syntax:

```
ALTER TABLE table_name
            ADD (Columnname_1  datatype,
            Columnname_2  datatype,
            …
            Columnname_n  datatype);
```

# DDL - SQL: ALTER (ADD, DROP, MODIFY)

## ALTER TABLE – DROP

DROP COLUMN is used to drop column in a table. Deleting the unwanted columns from the table.

Syntax:

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

# DDL - SQL: ALTER (ADD, DROP, MODIFY)

ALTER TABLE-MODIFY

It is used to modify the existing columns in a table. Multiple columns can also be modified at once.
*Syntax may vary slightly in different databases.
Syntax(Oracle,MySQL,MariaDB):

```
 ALTER TABLE table_name
MODIFY column_name column_type;
```

Syntax(SQL Server):

```
 ALTER TABLE table_name
ALTER COLUMN column_name column_type;
```

# DDL - SQL: ALTER (ADD, DROP, MODIFY)

Queries
Sample Table:

| Student | |
|---|---|
| **ROLL_NO** | **NAME** |
| 1 | Ram |
| 2 | Abhi |
| 3 | Rahul |
| 4 | Tanu |

# DDL - SQL: ALTER (ADD, DROP, MODIFY)

QUERY:

To ADD 2 columns AGE and COURSE to table Student.

```
ALTER TABLE Student ADD (AGE number(3),COURSE varchar(40));
```

OUTPUT:

| ROLL_NO | NAME | AGE | COURSE |
|---|---|---|---|
| 1 | Ram | | |
| 2 | Abhi | | |
| 3 | Rahul | | |
| 4 | Tanu | | |

# DDL - SQL: ALTER (ADD, DROP, MODIFY)

MODIFY column COURSE in table Student

```
ALTER TABLE Student MODIFY COURSE varchar(20);
```

After running the above query maximum size of Course Column is reduced to 20 from 40.

DROP column COURSE in table Student.

```
ALTER TABLE Student DROP COLUMN COURSE;
```

OUTPUT:

| ROLL_NO | NAME | AGE |
|---|---|---|
| 1 | Ram | |
| 2 | Abhi | |
| 3 | Rahul | |
| 4 | Tanu | |

# DDL - SQL: Comments

As is any programming languages comments matter a lot in SQL also. In this set we will learn about writing comments in any SQL snippet.
Comments can be written in the following three formats:

- Single line comments.
- Multi line comments
- In line comments

**Single line comments:** Comments starting and ending in a single line are considered as single line comments.
Line starting with '–' is a comment and will not be executed.
Syntax:

```
-- single line comment
-- another comment
SELECT * FROM Customers;
```

# DDL - SQL: Comments

**Multi line comments:** Comments starting in one line and ending in different line are considered as multi line comments. Line starting with ‘/*’ is considered as starting point of comment and are terminated when ‘*/’ is encountered.
Syntax:

```
/* multi line comment
another comment */
SELECT * FROM Customers;
```

In line comments: In line comments are an extension of multi line comments, comments can be stated in between the statements and are enclosed in between ‘/*’ and ‘*/’.
Syntax:

```
SELECT * FROM /* Customers; */
```

# DDL - SQL: Comments

**More examples:**

```
Multi line comment ->
/* SELECT * FROM Students;
SELECT * FROM STUDENT_DETAILS;
SELECT * FROM Orders; */
SELECT * FROM Articles;

In line comment ->
SELECT * FROM Students;
SELECT * FROM /* STUDENT_DETAILS;
SELECT * FROM Orders;
SELECT * FROM */ Articles;
```

# DDL - SQL: ALTER (RENAME)

Sometimes we may want to rename our table to give it a more relevant name. For this purpose we can use ALTER TABLE to rename the name of table.
*Syntax may vary in different databases.

**Syntax(Oracle,MySQL,MariaDB):**

```
ALTER TABLE table_name
RENAME TO new_table_name;
```

Columns can be also be given new name with the use of ALTER TABLE.
Syntax(Oracle):

```
ALTER TABLE table_name
RENAME COLUMN old_name TO new_name;
```

# DDL - SQL: ALTER (RENAME)

**Syntax(MySQL,MariaDB):**

```
ALTER TABLE table_name
CHANGE COLUMN old_name TO new_name;
```

Sample Table:

Student

| ROLL_NO | NAME | AGE |
|---|---|---|
| 1 | Ram | 20 |
| 2 | Abhi | 21 |
| 3 | Rahul | 22 |
| 4 | Tanu | 19 |

# DDL - SQL: ALTER (RENAME)

QUERY:

Change the name of column NAME to FIRST_NAME in table Student.

```
ALTER TABLE Student RENAME COLUMN NAME TO FIRST_NAME;
```

OUTPUT:

| ROLL_NO | FIRST_NAME | AGE |
|---|---|---|
| 1 | Ram | 20 |
| 2 | Abhi | 21 |
| 3 | Rahul | 22 |
| 4 | Tanu | 19 |

# DDL - SQL: ALTER (RENAME)

Change the name of the table Student to Student_Details

```
ALTER TABLE Student RENAME TO Student_Details;
```

OUTPUT:

Student_Details

| ROLL_NO | FIRST_NAME | AGE |
|---|---|---|
| 1 | Ram | 20 |
| 2 | Abhi | 21 |
| 3 | Rahul | 22 |
| 4 | Tanu | 19 |

# Congratulations!