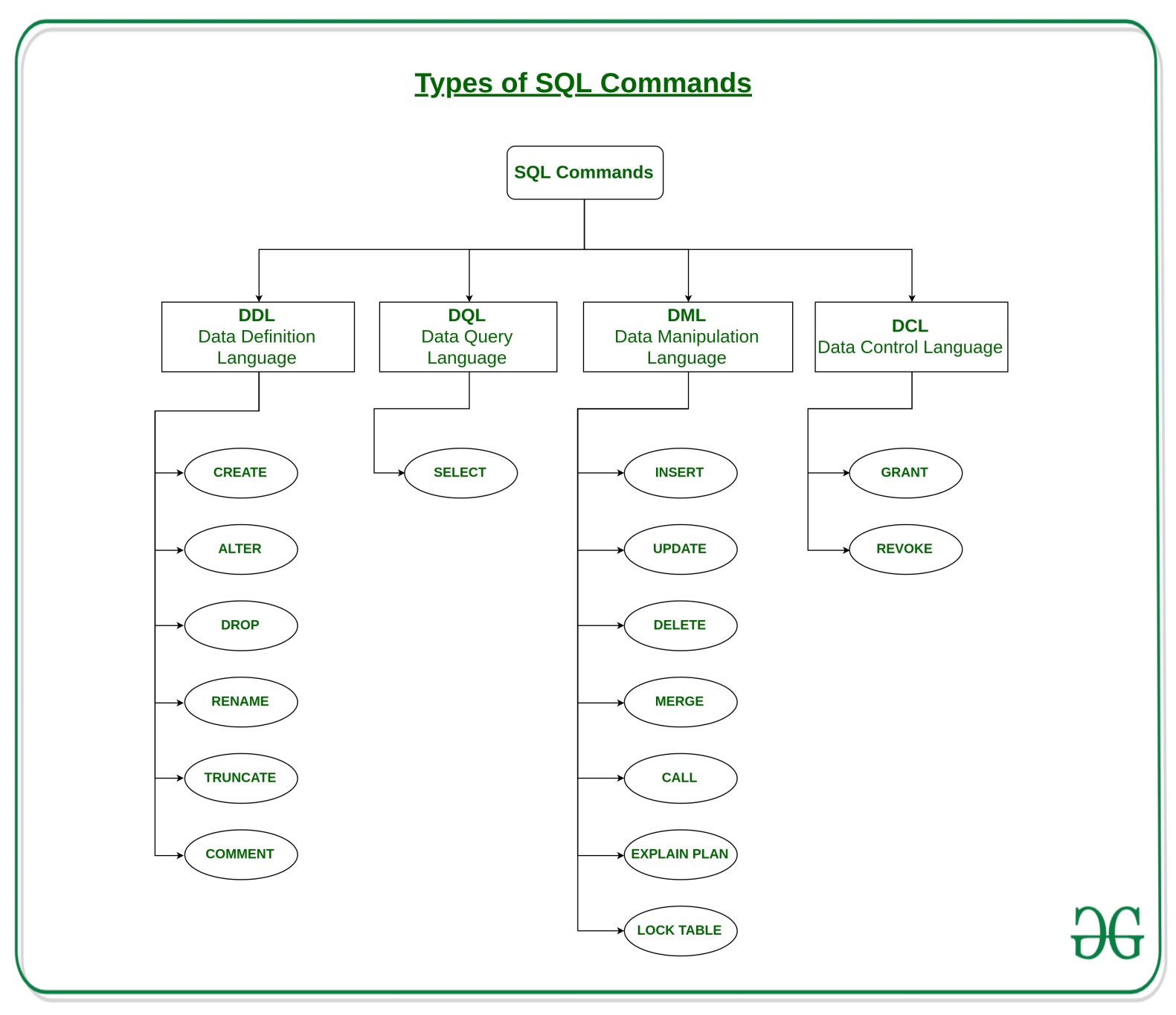
Structured Query Language(SQL) as we all know is the database language by the use of which we can perform certain operations on the existing database and also we can use this language to create a database. SQL uses certain commands like Create, Drop, Insert etc. to carry out the required tasks.

These SQL commands are mainly categorized into four categories as:

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

**Types of SQL Commands –**



**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.

Examples of DDL commands:

* [**CREATE**](https://www.geeksforgeeks.org/sql-create/) – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
* [**DROP**](https://www.geeksforgeeks.org/sql-drop-truncate/) – is used to delete objects from the database.
* [**ALTER**](https://www.geeksforgeeks.org/sql-alter-add-drop-modify/)-is used to alter the structure of the database.
* [**TRUNCATE**](https://www.geeksforgeeks.org/sql-drop-truncate/)–is used to remove all records from a table, including all spaces allocated for the records are removed.
* [**RENAME**](https://www.geeksforgeeks.org/sql-alter-rename/)–is used to rename an object existing in the database.

**CREATE COMMAND –**

* **How to Create a Table in Database –**

**Syntax for Creating a table –**

CREATE TABLE table\_name (  
    column1 datatype,  
    column2 datatype,  
    column3 datatype,  
   ....  
);

Where,

* The column parameters specify the names of the columns of the table.
* The datatype parameter specifies the type of data the column can hold (e.g. varchar, integer, date, etc.).

**Example –**

CREATE TABLE Persons (  
    PersonID int,  
    LastName varchar(255),  
    FirstName varchar(255),  
    Address varchar(255),  
    City varchar(255)  
);

* The PersonID column is of type int and will hold an integer.
* The LastName, FirstName, Address, and City columns are of type varchar and will hold characters, and the maximum length for these fields is 255 characters.

To view table use command – select \* from tablename

* **Create Table using another table in a database –**

**Syntax for Creating a table using another table -**

CREATE TABLE new\_table\_name AS  
    SELECT column1, column2,...  
    FROM existing\_table\_name  
    WHERE ....;

**Example –**

CREATE TABLE TestTable AS  
SELECT customername, contactname  
FROM customers;

* **Constraints use while creating a Table and its attributes in a database –**

The following constraints are commonly used in SQL:

* [**NOT NULL**](https://www.w3schools.com/sql/sql_notnull.asp) - Ensures that a column cannot have a NULL value
* [**UNIQUE**](https://www.w3schools.com/sql/sql_unique.asp) - Ensures that all values in a column are different
* [**PRIMARY KEY**](https://www.w3schools.com/sql/sql_primarykey.asp) - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
* [**FOREIGN KEY**](https://www.w3schools.com/sql/sql_foreignkey.asp) - Uniquely identifies a row/record in another table
* [**CHECK**](https://www.w3schools.com/sql/sql_check.asp) - Ensures that all values in a column satisfies a specific condition
* [**DEFAULT**](https://www.w3schools.com/sql/sql_default.asp) - Sets a default value for a column when no value is specified

**NOT NULL CONSTRAINTS -**

**Example for NOT NULL –**

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255) NOT NULL,  
    Age int  
);

**SQL NOT NULL on ALTER Command EXAMPLE -**

ALTER TABLE Persons  
MODIFY Age int NOT NULL;

**UNIQUE CONSTRAINTS -**

**Example for Unique Constraints –**

There are three ways in which unique constraints can be added –

**TYPE 1 -**

CREATE TABLE Persons (  
    ID int NOT NULL UNIQUE,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int  
);

**TYPE 2 -**

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    UNIQUE (ID)  
);

**TYPE 3 -**

\*\* If we need to do for more than one column

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    CONSTRAINT UC\_Person UNIQUE (ID,LastName)  
);

**SQL UNIQUE on ALTER Command EXAMPLE -**

ALTER TABLE Persons  
ADD UNIQUE (ID);

**PRIMARY CONSTRAINTS –**

**Example for primary –**

There are three ways in which unique constraints can be added –

**TYPE 1 -**

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    PRIMARY KEY (ID)  
);

**TYPE 2 -**

CREATE TABLE Persons (  
    ID int NOT NULL PRIMARY KEY,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int  
);

**TYPE 3 -**

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    CONSTRAINT PK\_Person PRIMARY KEY (ID,LastName)  
);

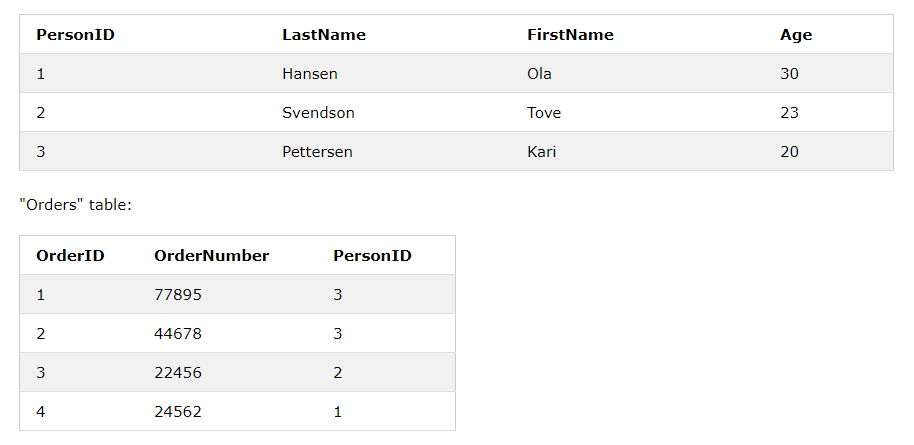
**SQL PRIMARY on ALTER Command EXAMPLE -**

ALTER TABLE Persons  
ADD PRIMARY KEY (ID);

**FOREIGN CONSTRAINTS –**

* A FOREIGN KEY is a key used to link two tables together.
* A FOREIGN KEY is a field (or collection of fields) in one table that refers to the PRIMARY KEY in another table.

Consider an example below -



* Notice that the "PersonID" column in the "Orders" table points to the "PersonID" column in the "Persons" table.
* The "PersonID" column in the "Persons" table is the PRIMARY KEY in the "Persons" table.
* The "PersonID" column in the "Orders" table is a FOREIGN KEY in the "Orders" table.

**Example for foreign –**

There are three ways in which unique constraints can be added –

**TYPE 1 -**

CREATE TABLE Orders (  
    OrderID int NOT NULL,  
    OrderNumber int NOT NULL,  
    PersonID int,  
    PRIMARY KEY (OrderID),  
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)  
);

**TYPE 2 -**

CREATE TABLE Orders (  
    OrderID int NOT NULL PRIMARY KEY,  
    OrderNumber int NOT NULL,  
    PersonID int FOREIGN KEY REFERENCES Persons(PersonID)  
);

**TYPE 3 -**

CREATE TABLE Orders (  
    OrderID int NOT NULL,  
    OrderNumber int NOT NULL,  
    PersonID int,  
    PRIMARY KEY (OrderID),  
    CONSTRAINT FK\_PersonOrder FOREIGN KEY (PersonID)  
    REFERENCES Persons(PersonID)  
);

**SQL FOREIGN on ALTER Command EXAMPLE -**

ALTER TABLE Orders  
ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);

**CHECK CONSTRAINTS –**

* The CHECK constraint is used to limit the value range that can be placed in a column.
* If you define a CHECK constraint on a single column it allows only certain values for this column.
* If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.

**EXAMPLE 1 -**

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    CHECK (Age>=18)  
);

**EXAMPLE 2 -**

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    City varchar(255),  
    CONSTRAINT CHK\_Person CHECK (Age>=18 AND City='Nagpur')  
);

**SQL CHECK on ALTER Command EXAMPLE -**

ALTER TABLE Persons  
ADD CHECK (Age>=18);

**DEFAULT CONSTRAINTS –**

* The DEFAULT constraint is used to provide a default value for a column.
* The default value will be added to all new records IF no other value is specified.

**EXAMPLE 1 -**

CREATE TABLE Persons (  
    ID int NOT NULL,  
    LastName varchar(255) NOT NULL,  
    FirstName varchar(255),  
    Age int,  
    City varchar(255) DEFAULT 'Nagpur'  
);

**EXAMPLE 2 -**

CREATE TABLE Orders (  
    ID int NOT NULL,  
    OrderNumber int NOT NULL,  
    OrderDate date DEFAULT GETDATE()  
);

**SQL DEFAULT on ALTER Command EXAMPLE -**

ALTER TABLE Persons  
ALTER City SET DEFAULT 'Nagpur';

**DROP COMMAND –**

* The SQL **DROP TABLE** statement is used to remove a table definition and all the data, indexes, triggers, constraints and permission specifications for that table.
* **NOTE** − You should be very careful while using this command because once a table is deleted then all the information available in that table will also be lost forever.

**Syntax for Dropping a table from a database –**

DROP TABLE table\_name;

**Example for Dropping a table from a database –**

DROP TABLE order;

**ALTER COMMAND –**

The SQL **ALTER TABLE** command is used to add, delete or modify columns in an existing table. You should also use the ALTER TABLE command to add and drop various constraints on an existing table.

**Syntax for Altering a table in a database –**

The basic syntax of an ALTER TABLE command to add a **New Column** in an existing table is as follows.

ALTER TABLE table\_name ADD column\_name datatype;

The basic syntax of an ALTER TABLE command to DROP COLUMN in an existing table is as follows.

ALTER TABLE table\_name DROP COLUMN column\_name;

The basic syntax of an ALTER TABLE command to change the DATA TYPE of a column in a table is as follows.

ALTER TABLE table\_name MODIFY COLUMN column\_name datatype;

**TRUNCATE COMMAND –**

A truncate SQL statement is used to remove all rows (complete data) from a table. It is similar to the DELETE statement with no WHERE clause.

**Syntax for truncating a table in a database –**

**TRUNCATE** **TABLE** table\_name;

**Example for truncating a table in a database –**

**TRUNCATE** **TABLE** Employee;

**RENAME COMMAND –**

**SQL RENAME TABLE** syntax is used to change the name of a table. Sometimes, we choose non-meaningful name for the table. So it is required to be changed.

Let's see the syntax to rename a table from the database.

**Syntax for renaming a table in a database –**

**ALTER** **TABLE** table\_name RENAME **TO** new\_table\_name;

**Example for renaming a table in a database –**

Let us take an example of a table named "STUDENTS", now due to some reason we want to change it into table name "ARTISTS".

**ALTER** **TABLE** STUDENTS RENAME **TO** ARTISTS;

**DQL – Data Query Language**

DML statements are used for performing queries on the data within schema objects. The purpose of DQL Command is to get some schema relation based on the query passed to it.

**Example of DQL:**

* [**SELECT**](https://www.geeksforgeeks.org/sql-select-clause/) – is used to retrieve data from a database.

EXERCISE –