**DDL** is Data Definition language.

Note: Different databases uses different keyword for certain operations. Always check documentation to avoid issues. Here, codes are tested on XAMPP

1. **Create**:
   1. *Create Database*: CREATE DATABASE databasename;
   2. *Replace Database*: CREATE OR REPLACE DATABASE databasename;
   3. *Create Table*: This has several versions. Also, we can add **Unique/Not NULL** **Constraints** in the definitions.

* we can create new table like this,

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

* we can create table using another table. Here, we can select all column, or specific columns. We can use conditions as well.

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

1. **Alter:**
   1. *Add Column:* To add a column to a table,

ALTER TABLE table\_name  
ADD COLUMN (

column\_name datatype,

----------

);

* 1. *Change/Modify Column:* This has different syntax in different databases, some databases use **ALTER** and some use **MODIFY.** We can change both column names and datatype here.

ALTER TABLE table\_name  
MODIFY COLUMN (

column\_name datatype,

----------

);

1. **Drop:**
   1. *Drop Database:* DROP DATABASE databasename/s;
   2. *Drop Table*: DROP TABLE table\_name;
   3. *Drop Column:*

ALTER TABLE table\_name  
DROP COLUMN column\_name;

1. **Truncate:** This is used to just delete the data of the table, not the table itself.

TRUNCATE TABLE table\_name;

1. **Rename:** 
   1. *Rename Database:* RENAME DATABASE*old\_database\_name*TO*new\_database\_name;*
   2. *Rename Table:* ALTER TABLE table\_name RENAME new\_table\_name
   3. *Rename Column:*

ALTER TABLE table\_name

CHANGE column\_name new\_column\_name datatype

**Constraints:** These are used to specify rules for table data. Below is a list of constraints.

**NOT NULL:** By Default, column can hold null values. We can change this by adding constraints. Below is example for creating and altering existing table.

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

**UNIQUE:**  Makes a column not allow duplicates. Adding Keyword **UNIQUE** similar to the example of **NOT NULL** above.

**PRIMARY KEY:** By default, NOT NULL and UNIQUE. A table can have only one primary key. It can consist of single or multiple columns. Below is example for both.

CONSTRAINT PK\_ID PRIMARY KEY (ID)

CONSTRAINT PK\_Person PRIMARY KEY (ID, LastName)

Note: Even though in second example primary key is made of two columns, it still one key (PK\_Person)

**FOREIGN KEY:** Foreign key refers to the primary key in another table. It is also knowns as referencing key.

* Foreign key can be NULL or Duplicate.
* To create a foreign key, primary key of parent table must be created first.
* A table can have multiple foreign keys.

CONSTRAINT PK\_ID

FOREIGN KEY (Child\_table\_foreign\_key\_column\_name)

REFERENCES parent\_table\_name(parent\_table\_primary\_key)

**CHECK:** It is used to limit the range of values that can be added in a column.

CONSTRAINT CHK\_Person CHECK (Age>=18 AND City='Sandnes')

**DEFAULT:** Used to set a default value for a column.

City varchar (255) DEFAULT 'Sandnes'

**CREATE INDEX:** Indexes are used when we need to retrieve data quickly, downside is it takes more time to update data.

**Composite Primary Key:** Composite key is combination of two or more columns in a table.

* CONSTRAINT PK\_Person PRIMARY KEY (First\_Name,Last\_Name)

**Self-Referencing Foreign Key:** Also known as recursive foreign key. Here foreign key and primary key is in the same table.

**DML** is data manipulation language.

1. **Insert:** We don’t have specify column names in case we are adding values for all the table. Otherwise, we have to specify column names, and insert values in right order

* INSERT INTO table\_name (column1, column2, column3, ...)  
  VALUES (value1, value2, value3, ...);
* INSERT INTO table\_name  
  VALUES (value1, value2, value3, ...);

1. **Select:** Basic selection command. In case of string, we must use quotation.

* SELECT column1, column2, ...  
  FROM table\_name

WHERE condition

1. **Update:** Basic update command. Important: if we don’t use where to specify desired records, whole table will be updated and this process is not reversable.

* UPDATE table\_name  
  SET column1 = value1, column2 = value2, ...  
  WHERE condition;

1. **Delete:** Basic delete command.

* DELETE FROM table\_name WHERE condition;

**Conditions in SQL:**

* **logical operators** such as AND, OR, NOT
* **Mathematical operators** such as **<; >; <=; >=; =; ! =; <> ;+;−; \*; ~ :%**