**Intro to DDL, DML Commands**

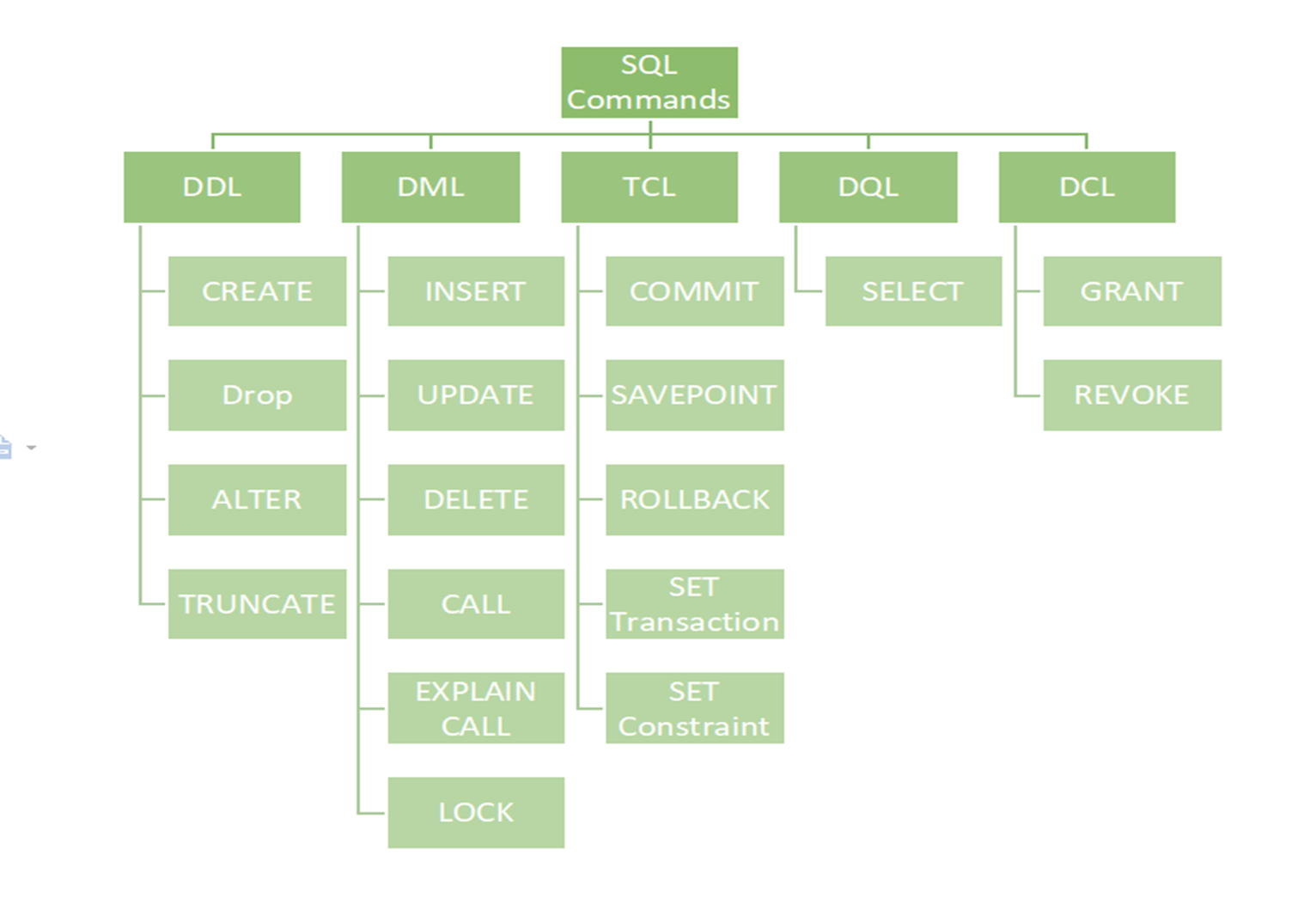
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

For this lecture, we will be using **MySQL Workbench** instead of BigQuery.

What is MySQL?

MySQL is an open-source relational database management system developed by Oracle that is based on structured query language (SQL).

How to get started with MySQL - [MySQL Installation and Setup.pdf](https://drive.google.com/file/d/1gJ2W4HFY6YxYMX1xtjOyKOefW93WoO-y/view?usp=drive_link)



* **DDL** - Data Definition Language
* **DML** - Data Manipulation Language
* **TCL** - Transaction Control Language
* **DQL** - Data Query Language - most important
* **DCL** - Data Control Language

**What is DDL?**

* Data Definition Language (DDL) is used to define the database schema. It includes commands such as CREATE, ALTER, DROP, and TRUNCATE.

**What is DML?**

* Data Manipulation Language (DML), on the other hand, is used to manipulate data within the database. It includes commands such as SELECT, INSERT, UPDATE, and DELETE.

Let's start by discussing the **DDL commands**.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**CREATE command:**

The CREATE command is used to **create** a new database, table, view, or stored procedure in MySQL.

The syntax for the CREATE command is as follows:

* CREATE [DATABASE | TABLE | VIEW | PROCEDURE] object\_name;

Let’s start by creating a database named “**new\_db**”.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

How to create a database in MySQL?

**Syntax:**

CREATE DATABASE database\_name;

**Query:**

CREATE DATABASE new\_db;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

How to use a database in MySQL?

**Syntax:**

USE database\_name;

**Query:**

USE new\_db;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now, let’s create a new table named ‘**customers**’.

Our table will contain the following column:

* ID
* first\_name
* last\_name
* age
* gender
* phone\_no
* email\_id
* dob
* address

**Syntax:**

CREATE TABLE table\_name (

Column1 data\_type <CONSTRAINT(s)>,

Column2 data\_type <CONSTRAINT(s)>,

Column3 data\_type <CONSTRAINT(s)>

);

**Query:**

CREATE TABLE customers (

ID INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(20) NOT NULL,

last\_name VARCHAR(20),

age INT CHECK(age>=18),

gender ENUM(“M”, “F”),

phone\_no CHAR(10) NOT NULL UNIQUE,

email\_id VARCHAR(30),

dob DATE,

address VARCHAR(100)

);

You can view the newly created “**new\_db**” database and the “**customers**” table under the **SCHEMA** section in the MySQL WOrkbench.

You’ll have to click the  icon for the changes to be reflected.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Constraints**

SQL constraints are used to specify rules for the data in a table.

* **UNIQUE** - Ensures that all values in a column are unique.
* **NOT NULL** - Ensures that a column cannot have a NULL value.
* **PRIMARY KEY** -
  + A combination of NOT NULL and UNIQUE.
  + Uniquely identifies each row in a table
* **FOREIGN KEY** -
  + Prevents invalid data from being inserted into the foreign key column,
  + because it has to be one of the values contained in the parent table.
* **CHECK** - Ensures that the values in a column satisfies a specific condition.
* **DEFAULT** - Sets a default value for a column if no value is specified.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Data Types**

Here INT, DATE, VARCHAR, ENUM... These are all SQL data types.

They’re used to define the type of data which is stored in a column.

There are three main data types in MySQL:

1. string,
2. numeric, and
3. date & time.

You can refer to this link to showcase different MySQL data types. <https://www.w3schools.com/sql/sql_datatypes.asp>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**String → Char(size), Varchar(size), Enum(val1, val2, val3…)**

* **Char** is Fixed Length Character String
  + Eg. for Char(3) data type, a valid data entry could be IND, AUS, USA

Whereas inputs like INDIA, IN will throw an error.

* **Varchar** is Variable Length Character String
  + Eg. for Varchar(5) data type, a valid entry could be IN, IND, INDIA

Whereas inputs like AMERICA, AUSTRALIA will throw an error.

* **Enum** is a data type that allows you to have only one value, chosen from a list of possible values.
  + Eg. for Enum(“Yes”, “No”), valid data entries could either be or
  + If we try to insert a value which is not in the list, a blank value will be inserted.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Numeric → Int, Float(size, d), Bool**

* **Int** is for storing an integer value.
  + Eg. 2, 0, -10, etc.
* **Float**(size, d) is for storing a floating point number.
  + The total number of digits is specified in size.
  + The number of digits after the decimal point is specified in the d parameter.
* **Bool** is used to store a True/False value.
  + Zero is considered as false
  + Non-zero values are considered as true.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Date & Time** **→ Date, Time, Datetime, Timestamp**

* **Date** data type is used for storing a date in the format YYYY-MM-DD.
  + Eg. 2023-05-17
* **Time** data type is used for storing a time in the format hh:mm:ss.
  + Eg. 03:07:00
* **Datetime** is used for storing a combination of date & time in the format YYYY-MM-DD hh:mm:ss.
  + Eg. 2023-05-17 03:07:00
* **Timestamp** values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ALTER command:**

The ALTER command is used to **modify** an existing database, table, view, or stored procedure in MySQL.

The syntax for the ALTER command is as follows:

* ALTER [DATABASE | TABLE | VIEW | PROCEDURE] object\_name [modification];

The ALTER command can be used in various ways to add different modifications to our database or table.

For example,

* Adding new column(s) to a table,
* Changing the data type of a column,
* Renaming an existing column,
* Adding constraints to a column,
* Deleting a column,
* Renaming a table, etc.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Add a new column named ‘is\_active’ to the “customers” table.

We can do this using the **ADD** keyword along with the ALTER command.

**Syntax:**

ALTER TABLE table\_name

**ADD** column\_name data\_type CONSTRAINT(S);

**Query:**

ALTER TABLE customers

ADD is\_active VARCHAR(10);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Change the data type of the ‘is\_active’ column to INT.

We can do this using the **MODIFY** keyword along with the ALTER command.

**Syntax:**

ALTER TABLE table\_name

**MODIFY** column\_name new\_data\_type;

**Query:**

ALTER TABLE customers

MODIFY is\_active INT;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Add constraint to the ‘is\_active’ column.

We can do this using the **ADD CONSTRAINT** keyword along with the ALTER command.

**Syntax:**

ALTER TABLE table\_name

**ADD CONSTRAINT** constraint\_name CONSTRAINT(column1, column2…);

**Query:**

ALTER TABLE customers

ADD CONSTRAINT con CHECK(is\_active IN(1,0);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Rename the ‘ID’ column to ‘cust\_id’.

We can do this using the **RENAME** **COLUMN** keyword along with the ALTER command.

**Syntax:**

ALTER TABLE table\_name

**RENAME COLUMN** old\_col\_name TO new\_col\_name;

**Query:**

ALTER TABLE customers

RENAME COLUMN ID TO cust\_id;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Delete the ‘address’ column from the “customers” table.

We can do this using the **DROP** **COLUMN** keyword along with the ALTER command.

**Syntax:**

ALTER TABLE table\_name

**DROP** **COLUMN** column\_name;

**Query:**

ALTER TABLE customers

DROP COLUMN address;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Rename the “customers” table to “cust\_info”.

**Method 1:**

We can do this using the **RENAME TO** keyword along with the ALTER command.

**Syntax:**

ALTER TABLE table\_name

**RENAME TO** new\_table\_name;

**Query:**

ALTER TABLE customers

RENAME TO cust\_info;

**Method 2:**

**Query:**

RENAME TABLE customers

TO cust\_info;

**RENAME TABLE**, unlike the ALTER command, can rename multiple tables within a single statement.

**E.g.**

**RENAME TABLE**

old\_table1 TO new\_table1,

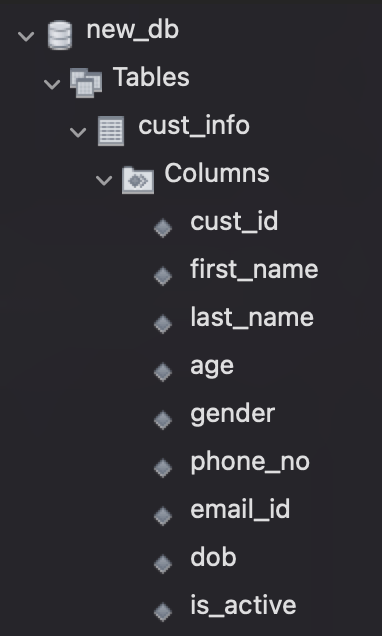
old\_table2 TO new\_table2,

old\_table3 TO new\_table3;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now, after adding these modifications to the table, the schema for our **“cust\_info”** table looks something like this…

Please note that till’ this point, we haven’t inserted a single record into our table.



Now, let's move on to the **DML commands**.

We’ll get back to the **TRUNCATE** and **DROP** commands later.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**INSERT command:**

The INSERT command is used to **insert** data into a table in MySQL.

**Syntax:**

**INSERT INTO** table\_name (column1, column2, ...)

**VALUES** (value1, value2, ...);

**Note:** We can omit specifying the column names if we intend to add values into all the columns present in the table.

Insert data into the “cust\_info” table that we created.

**Query:**

INSERT INTO cust\_info

VALUES

(101, 'John', 'Doe', 25, 'M', '8245639875', 'john@gmail.com', '1998-05-10', 1),

(102, 'Jane', 'Smith', 32, 'F', '1098476253', 'jane@gmail.com', '1991-09-22', 1),

(103, 'David', 'Johnson', 42, 'M', '7562314098', 'david@yahoo.com', '1979-07-15', 0),

(104, 'Sarah', 'Williams', 28, 'F', '3089576124', 'sarah@gmail.com', '1995-12-03', 1),

(105, 'Michael', 'Brown', 37, 'M', '9650138742', 'michael@gmail.com', '1986-03-27', 1),

(106, 'Emily', 'Davis', 29, 'F', '4721398056', 'emily@yahoo.com', '1994-11-08', 0),

(107, 'Daniel', 'Anderson', 31, 'M', '6214789305', 'daniel@gmail.com', '1992-08-19', 1),

(108, 'Olivia', 'Taylor', 24, 'F', '8356912074', 'olivia@gmail.com', '1999-02-14', 1),

(109, 'Matthew', 'Wilson', 36, 'M', '2034957168', '[matthew@gmail.com](mailto:matthew@gmail.com)', '1987-09-01', 0),

(110, 'Sophia', 'Martin', 27, 'F', '5893642701', 'sophia@yahoo.com', '1996-06-12', 1);

You can use the following query to view the data that we inserted into the “customer\_info” table.

SELECT \* FROM cust\_info;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**UPDATE:**

The UPDATE command is used to **modify** existing records in a table in MySQL.

**Syntax:**

**UPDATE** table\_name

**SET** column1 = value1, column2 = value2, …

WHERE condition;

For the customer having ‘cust\_id’ 101, update the ‘phone\_no’ to 9305884176.

**Query:**

UPDATE cust\_info

SET phone\_no = 9305884176

WHERE cust\_id = 101;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DELETE:**

The DELETE command is used to **delete** data from a table in MySQL.

**Syntax:**

**DELETE** FROM table\_name

WHERE condition;

Delete the record for a customer having ‘cust\_id’ as 108.

**Query:**

DELETE FROM cust\_info

WHERE cust\_id = 108;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Coming back to the DDL commands that we left earlier…

**TRUNCATE command:**

The TRUNCATE command is used to **clear** an existing table in MySQL.

The syntax for the TRUNCATE command is as follows:

* TRUNCATE TABLE table\_name;

For example, if we wanted to **delete all the records** fromthe **“cust\_info”** table -

**Query:**

TRUNCATE TABLE cust\_info;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DROP command:**

The DROP command is used to **delete** an existing database, table, view, or stored procedure in MySQL.

The syntax for the DROP command is as follows:

* DROP [DATABASE | TABLE | VIEW | PROCEDURE] object\_name;

For example, to drop the **“cust\_info”** table -

**Syntax:**

DROP TABLE table\_name;

**Query:**

DROP TABLE cust\_info;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**TRUNCATE vs DROP vs DELETE**

| TRUNCATE | DROP | DELETE |
| --- | --- | --- |
| It is a **DDL** command | It is a **DDL** command | It is a **DML** command |
| Used to delete all the records from a table leaving only the columns. | Used to drop a table or even a database. | Used to delete one or more specific records from a table. |
| **TRUNCATE** TABLE table\_name; | **DROP** TABLE table\_name; | **DELETE** FROM table\_name WHERE condition; |