

## MySQL-SQL

### WHAT IS SQL?

1. SQL stands for Structured Query Language.
2. Used for managing and manipulating relational databases.
3. SQL lets you access and manipulate databases.
4. SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987.

### WHAT CAN SQL DO?

1. SQL can execute queries against a database.
2. SQL can retrieve data from a database.
3. SQL can insert records in a database.
4. SQL can update records in a database.
5. SQL can delete records from a database.
6. SQL can create new databases.
7. SQL can create new tables in a database.
8. SQL can create stored procedures in a database.
9. SQL can create views in a database.
10. SQL can set permissions on tables, procedures, and views.

### LIST OF WELL KNOWN RELATIONAL DATABASE MANAGEMENT SYSTEMS

1. MySQL
2. PostgreSQL
3. Oracle Database
4. Microsoft SQL Server
5. SQLite
6. IBM Db2
7. MariaDB

## CASE SENSITIVE OR NOT?

- KEYWORDS AND IDENTIFIERS ARE CASE INSENSITIVE LITERALS ARE CASE SENSITIVE.

## WHAT DO YOU MEAN BY DBMS? WHAT ARE ITS DIFFERENT TYPES?

Database is a structured collection of data.

A Database Management System (DBMS) is a software application that interacts with the user, applications and the database itself to capture and analyse data.

A DBMS allows a user to interact with the database using query language such as SQL. The data stored in the database can be modified, retrieved and deleted and can be of any type like strings, numbers, images etc.

## THERE ARE TWO TYPES OF DBMS:

1. **Relational Database Management System:** The data is stored in relations (tables). Example – MySQL, Oracle SQL.
2. **Non-Relational Database Management System:** There is no concept of relations, tuples and attributes. Example – Mongo

## WHAT ARE THE DIFFERENT SUBSETS OF SQL?

The standard SQL commands to interact with relational databases are CREATE, SELECT, INSERT, UPDATE, DELETE and DROP. These commands can be classified into the following groups based on their nature.

## 1. DDL - Data Definition Language

	Command & Description
1	<b>CREATE</b> Creates a new table, a view of a table, or other object in the database.
2	<b>ALTER</b> Modifies an existing database object, such as a table.
3	<b>DROP</b> Deletes an entire table, a view of a table or other objects in the database.

## 2. DML - Data Manipulation Language

	Command & Description
1	<b>SELECT</b> Retrieves certain records from one or more tables.
2	<b>INSERT</b> Creates a record.
3	<b>UPDATE</b> Modifies records.
4	<b>DELETE</b> Deletes records.

## 3. DCL - Data Control Language

	Command & Description
1	<b>GRANT</b> Gives a privilege to user.
2	<b>REVOKE</b> Takes back privileges granted from user.

## 4. DQL - Data Query Language

	Command & Description
1	<b>SELECT</b> The SELECT statement is used to retrieve data from one or more tables.
2	<b>DISTINCT</b> The DISTINCT keyword is used with SELECT to retrieve unique values from a specified column or a combination of columns.
3	<b>FROM</b> The FROM clause specifies the table or tables from which you want to retrieve data.

4	<b>WHERE</b>  The WHERE clause is used to filter rows based on a specified condition. It allows you to retrieve only the rows that meet the criteria you specify.
5	<b>ORDER BY</b>  The ORDER BY clause is used to sort the result set in ascending (ASC) or descending (DESC) order based on one or more columns.
6	<b>GROUP BY</b>  The GROUP BY clause is used to group rows with the same values in one or more columns into summary rows.
7	<b>HAVING</b>  The HAVING clause is used to filter the results of a GROUP BY query based on a condition applied to the aggregated values.

### WHAT DO YOU MEAN BY TABLE AND FIELD IN SQL?

A table refers to a collection of data in an organised manner in form of rows and columns. A field refers to the number of columns in a table. For example:

**Table:** StudentInformation

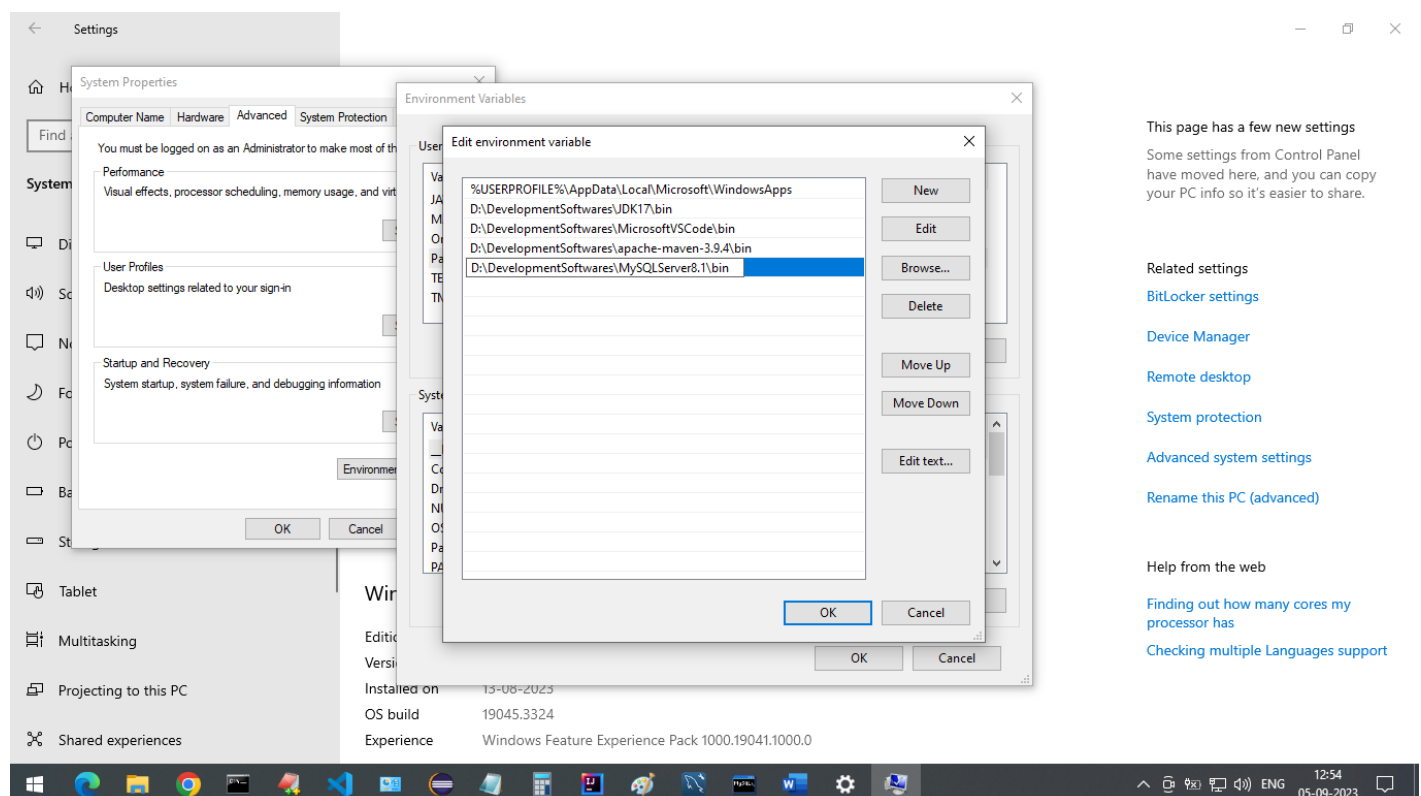
**Field:** StudentId, StudentName, StudentMarks

## Please follow the link to Learn how download and install MySQL Database and MySQL Workbench

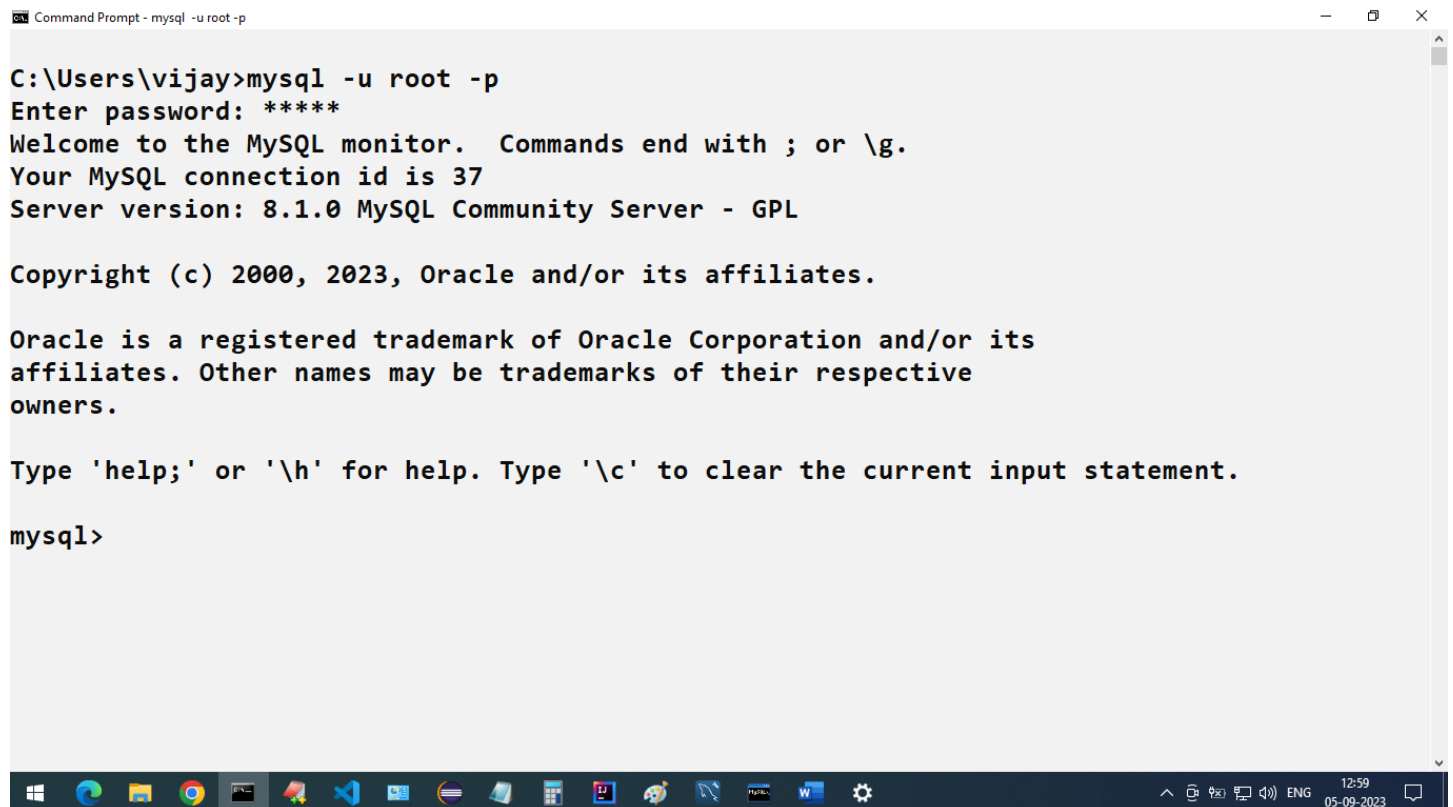
<https://rb.gy/3hcwf>

**Note:** If you are getting error while installing MySQL Server and MySQL Workbench like 'MySQL Workbench installer requires Visual C++ 2015' then follow [https://aka.ms/vs/17/release/vc\\_redist.x64.exe](https://aka.ms/vs/17/release/vc_redist.x64.exe) this link and download and install this piece of software.

## To Access the SQL Prompt from the windows command Line client set the path



## HOW TO LOGIN WITH A PARTICULAR USER FROM THE WINDOWS CMD PROMPT?

A screenshot of a Windows Command Prompt window titled "Command Prompt - mysql -u root -p". The window shows the execution of the command 'mysql -u root -p'. The prompt asks for a password, which is entered as '\*\*\*\*\*'. The MySQL monitor then displays a welcome message, the connection ID (37), the server version (8.1.0 MySQL Community Server - GPL), and copyright information. It also provides instructions on how to use help and clear the input. The prompt ends with 'mysql>'. The Windows taskbar is visible at the bottom with various application icons and a system clock showing 12:59 on 05-09-2023.

```
Command Prompt - mysql -u root -p

C:\Users\vijay>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 8.1.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

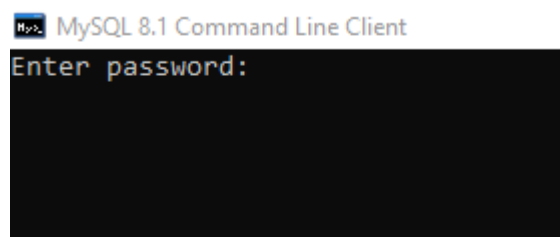
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

## HOW TO CHANGE USER IN THE MYSQL COMMAND LINE CLIENT?

**Note:** By default when you launch, you will be asked to enter the password for the root user.



Later you can change the user with the following command.

```
SYSTEM mysql -u vijay -p;
```

```
Enter password: *****
```

Alternatively you can use

```
\! mysql -u vijay -p
```

```
Enter password: *****
```

**Note:** In the Windows Command Prompt, you can log in with a specific user at the beginning. However, if you want to change the user, you can use the same command.

```
mysql> SYSTEM mysql -u vijay -p;  
Enter password: *****
```

### HOW TO CLOSE MYSQL COMMAND LINE CLIENT AS WELL AS TO EXIT FROM THE MYSQL PROMPT FROM WINDOWS COMMAND PROMPT?

**EXIT**

### HOW TO DISPLAY THE CURRENT USER?

You can use the USER() function to retrieve the current user. The USER() function returns the current user name and host name combination that the server used to authenticate the current client.

```
SELECT USER();
```

```
+-----+
```

```
| USER()      |
```

```
+-----+
```

```
| root@localhost |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

### HOW TO DISPLAY ALL THE DATABASES?

**SHOW DATABASES** command to get list of databases. Run the following query to show list of databases.



## FSD Training Program

**SHOW DATABASES;**

```
+-----+
| Database          |
+-----+
| information_schema |
| mysql             |
| mysql_notes       |
| performance_schema|
| student_tracker   |
| sys               |
+-----+
```

### HOW TO CREATE A NEW DATABASE?

**CREATE DATABASE MYSQL\_NOTES;**

**Query OK, 1 row affected (0.01 sec)**

**SHOW DATABASES;**

```
+-----+
| Database          |
+-----+
| information_schema |
| mysql             |
| mysql_notes       |
| performance_schema|
| student_tracker   |
| sys               |
+-----+
```

## HOW TO DELETE A DATABASE?

```
DROP DATABASE MYSQL_NOTES;
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
SHOW DATABASES;
```

```
+-----+
| Database                |
+-----+
| information_schema      |
| mysql                   |
| performance_schema     |
| student_tracker        |
| sys                     |
+-----+
```

## HOW TO SET OR SELECT A DATABASE?

- Before doing anything first we need to connect to a database.

```
USE MYSQL_NOTES;
```

```
Database changed
```

### HOW TO CHECK CURRENTLY WHICH DATABASE YOU ARE IN?

```
SELECT DATABASE();
```

```
+-----+
```

```
| DATABASE() |
```

```
+-----+
```

```
| mysql_notes |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

### HOW TO CREATE A NEW USER?

```
CREATE USER 'new_user'@'localhost' IDENTIFIED BY  
'password';
```

`new_user` is the name we've given to our new user account and the IDENTIFIED BY 'password' section sets a passcode for this user. You can replace these values with your own, inside the quotation marks.

In order to grant all privileges of the database for a newly created user, execute the following command:

```
GRANT ALL PRIVILEGES ON * . * TO 'new_user'@'localhost';
```

For changes to take effect immediately flush these privileges by typing in the command:

```
FLUSH PRIVILEGES;
```

### HOW TO DISPLAY ALL THE USERS FROM A DATABASE?

```
SELECT user, host FROM mysql.user;
```

In the above query `mysql` is the database.

### IS IT MANDATORY TO KEEP SEMICOLON AFTER SQL STATEMENTS?

1. Some database systems require a semicolon at the end of each SQL statement.
2. Semicolon is the standard way to separate each SQL statement in database systems that allow more than one SQL statement to be executed in the same call to the server.

### HOW TO CREATE A NEW USER WITH PASSWORD?

```
CREATE USER 'manager'@'localhost' IDENTIFIED BY 'admin';  
CREATE USER 'vijay'@'localhost' IDENTIFIED BY 'admin';
```

### HOW TO DROP EXISTING USER?

```
DROP USER 'manager'@'localhost';  
DROP USER 'vijay'@'localhost';
```

### HOW TO GRANT ALL PRIVILEGES TO THE USER?

```
GRANT ALL PRIVILEGES ON *.* TO 'vijay'@'localhost';
```

### HOW TO CHECK CURRENT USER PRIVILEGES?

```
SHOW GRANTS FOR 'root'@'localhost';
```

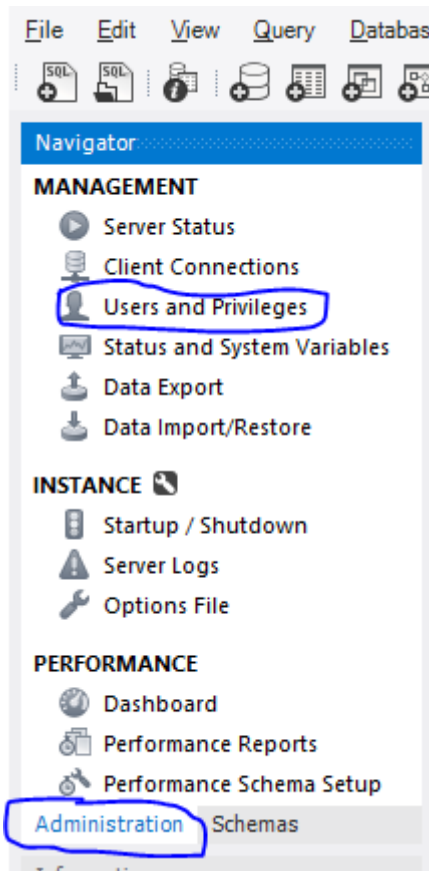
### HOW TO FOR CHANGES TO TAKE EFFECT IMMEDIATELY?

```
FLUSH PRIVILEGES;
```

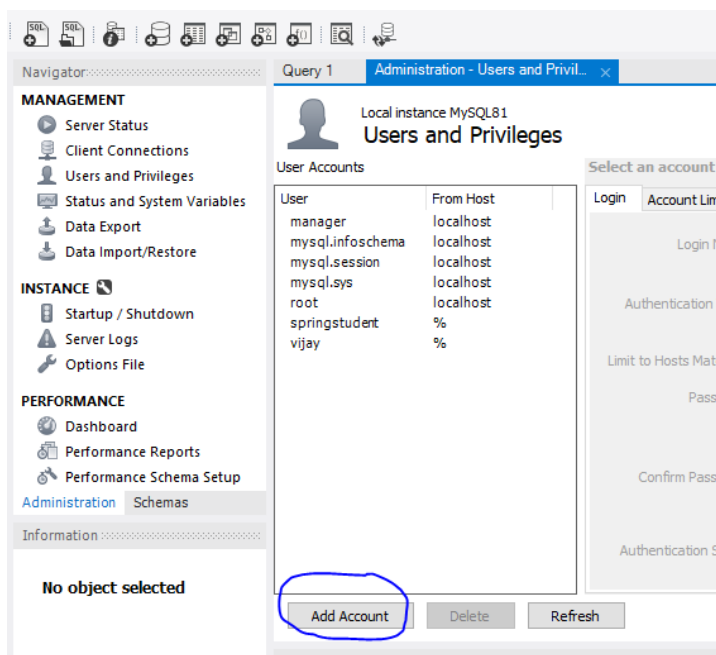
**Note:** Starting from MySQL 5.7.3, the FLUSH PRIVILEGES; statement is no longer strictly required after executing GRANT or REVOKE statements. The server automatically reloads the grant tables in these cases.

## HOW TO CREATE A NEW USER IN THE MYSQL WORKBENCH?

1. Log in to any connection
2. Click on Administration on the left hand side



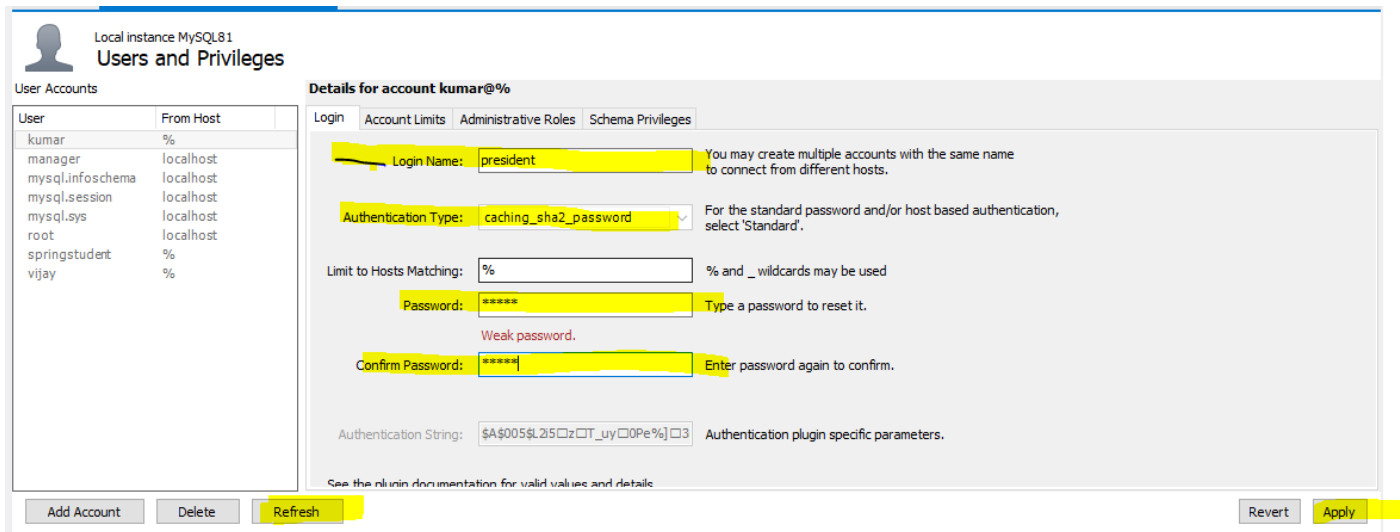
3. Click on Users and Privileges
4. Click on Add account to create a new account



5. Fill in the deatails

# FSD Training Program

**Note:** Authentication type should be same as other accounts(check for root)



The screenshot shows the 'Users and Privileges' window for 'Local instance MySQL81'. On the left, a table lists user accounts. On the right, the 'Details for account kumar@%' are shown with tabs for Login, Account Limits, Administrative Roles, and Schema Privileges. The 'Login' tab is active, showing fields for Login Name, Authentication Type, Password, Confirm Password, and Authentication String. The 'Apply' button is highlighted in yellow.

User	From Host
kumar	%
manager	localhost
mysql.infoschema	localhost
mysql.session	localhost
mysql.sys	localhost
root	localhost
springstudent	%
vijay	%

**Details for account kumar@%**

Login:  You may create multiple accounts with the same name to connect from different hosts.

Authentication Type:  For the standard password and/or host based authentication, select 'Standard'.

Limit to Hosts Matching:  % and \_ wildcards may be used

Password:  Type a password to reset it.

Weak password.

Confirm Password:  Enter password again to confirm.

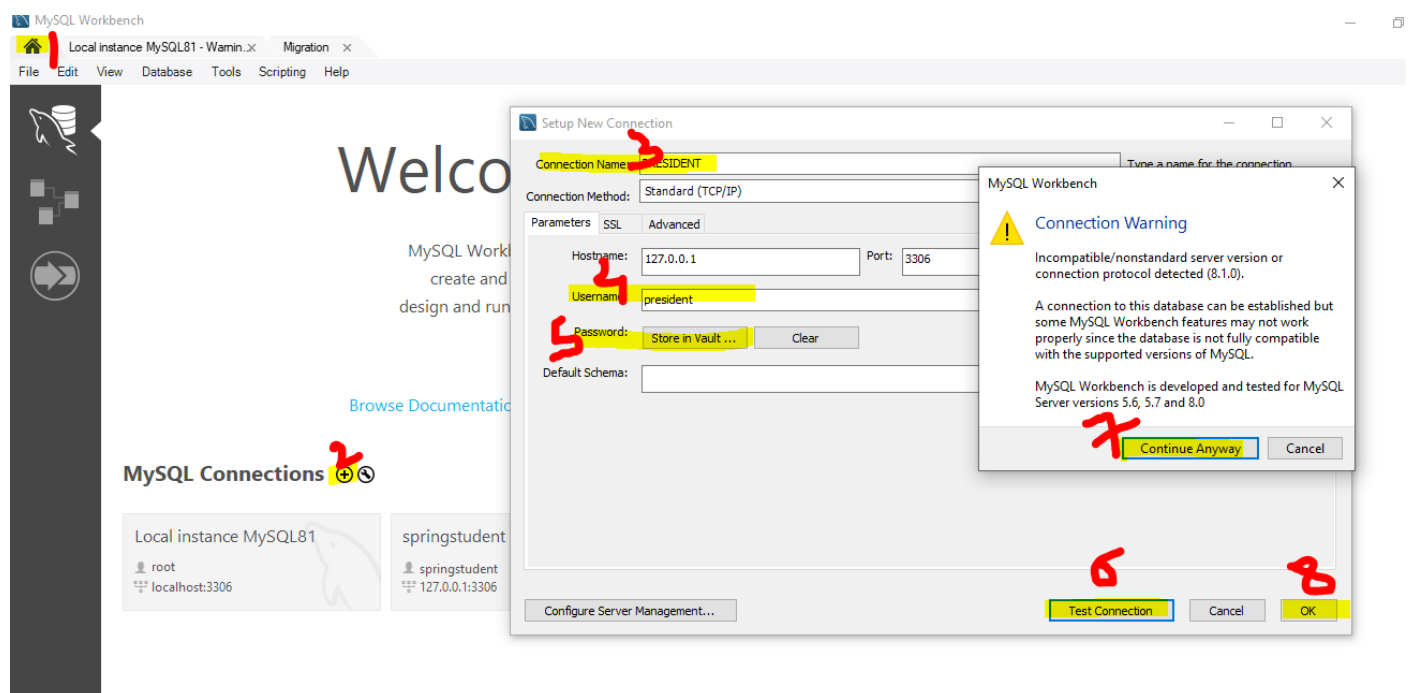
Authentication String:  Authentication plugin specific parameters.

See the plugin documentation for valid values and details

Buttons: Add Account, Delete, Refresh, Revert, Apply

6. Click on Apply and Refresh

## HOW TO ADD A NEW CONNECTION TO THE MYSQL WORKBENCH HOME?



The screenshot shows the MySQL Workbench interface with the 'Setup New Connection' dialog box open. The dialog has tabs for Parameters, SSL, and Advanced. The 'Parameters' tab is active, showing fields for Connection Name, Hostname, Port, Username, Password, and Default Schema. A 'Connection Warning' dialog box is also open, displaying a warning about incompatible/nonstandard server version or connection protocol detected (8.1.0). The 'Test Connection' button is highlighted in yellow.

MySQL Workbench

Local instance MySQL81 - Wamin.x Migration x

File Edit View Database Tools Scripting Help

Welcome to MySQL Workbench  
create and design and run

Browse Documentation

MySQL Connections

Local instance MySQL81  
root localhost:3306

springstudent  
springstudent 127.0.0.1:3306

Setup New Connection

Connection Name:

Connection Method: Standard (TCP/IP)

Parameters SSL Advanced

Hostname:  Port:

Username:

Password:  Clear

Default Schema:

Configure Server Management...

Test Connection Cancel OK

MySQL Workbench

Connection Warning

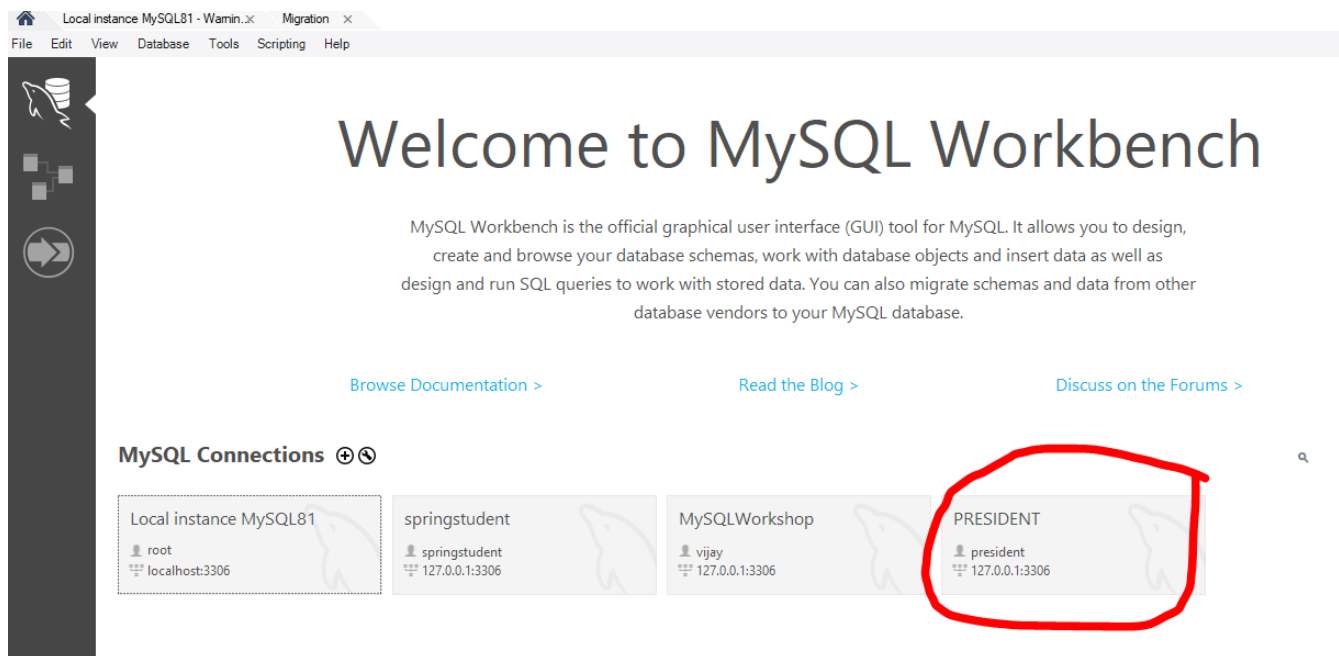
Incompatible/nonstandard server version or connection protocol detected (8.1.0).

A connection to this database can be established but some MySQL Workbench features may not work properly since the database is not fully compatible with the supported versions of MySQL.

MySQL Workbench is developed and tested for MySQL Server versions 5.6, 5.7 and 8.0

Continue Anyway Cancel

# FSD Training Program



**Note:** While creating the connection the user must be available(created already). Password is the user password that you have given at the time of creating a user.

## HOW TO CLEAR THE SCREEN IN MYSQL?

```
\! cls
```

## HOW TO CREATE A SIMPLE TABLE?

```
CREATE TABLE STUDENT (ID INTEGER, FIRST_NAME VARCHAR(90),  
AGE INTEGER, COURSE VARCHAR(10));
```

Query OK, 0 rows affected (0.03 sec)

- INTEGER is a data type synonym for INT.
- You can use both INT and INTEGER datatype to specify number types.

## HOW TO INSERT RECORDS TO THE TABLE?

```
INSERT INTO STUDENT VALUES (101, 'ARUN', 20, 'CSE');
```

```
INSERT INTO STUDENT VALUES (102, 'BHAVESH', 21, 'ISE');
```

```
INSERT INTO STUDENT VALUES (103, 'CHAITANYA', 22, 'ECE');
```

```
INSERT INTO STUDENT VALUES (104, 'DEEPIKA', 23, 'MECH');
```

## FSD Training Program

```
INSERT INTO STUDENT VALUES (105, 'DHANUSH', 24, 'DS');
INSERT INTO STUDENT VALUES (106, 'EKTA', 25, 'AI');
INSERT INTO STUDENT VALUES (107, 'GAURAV', 26, 'ARCH');
INSERT INTO STUDENT VALUES (108, 'HARSHITA', 27,
'CHEMICAL');
INSERT INTO STUDENT VALUES (109, 'ISHAAN', 28, 'CIVIL');
INSERT INTO STUDENT VALUES (110, 'JANU', 29, 'EEE');
```

### HOW TO DISPLAY ALL THE RECORDS WITH ALL THE COLUMNS?

```
SELECT * FROM STUDENT;
```

```
+-----+-----+-----+-----+
| ID    | FIRST_NAME | AGE  | COURSE |
+-----+-----+-----+-----+
| 101   | ARUN       | 20   | CSE    |
| 102   | BHAVESH    | 21   | ISE    |
| 103   | CHAITANYA  | 22   | ECE    |
| 104   | DEEPIKA    | 23   | MECH   |
| 105   | DHANUSH    | 24   | DS     |
| 106   | EKTA       | 25   | AI     |
| 107   | GAURAV     | 26   | ARCH   |
| 108   | HARSHITA   | 27   | CHEMICAL |
| 109   | ISHAAN     | 28   | CIVIL  |
| 110   | JANU       | 29   | EEE    |
+-----+-----+-----+-----+
```

```
10 rows in set (0.00 sec)
```



## CAN WE INSERT NULL VALUES TO THE COLUMNS?

- By default, columns will be allowing duplicate values.

```
INSERT INTO STUDENT VALUES (110, 'JANU', 29, 'EEE');
```

```
Query OK, 1 row affected (0.00 sec)
```

```
SELECT * FROM STUDENT;
```

ID	FIRST_NAME	AGE	COURSE
101	ARUN	20	CSE
102	BHAVESH	21	ISE
103	CHAITANYA	22	ECE
104	DEEPIKA	23	MECH
105	DHANUSH	24	DS
106	EKTA	25	AI
107	GAURAV	26	ARCH
108	HARSHITA	27	CHEMICAL
109	ISHAAN	28	CIVIL
110	JANU	29	EEE
110	JANU	29	EEE

```
11 rows in set (0.00 sec)
```

- By default, columns will be allowing 'null' values.
- In MySQL, NULL represents an unknown or missing value in a database table.

```
INSERT INTO STUDENT(ID, FIRST_NAME) VALUES(111, 'PRANAV');
```

Query OK, 1 row affected (0.01 sec)

```
SELECT * FROM STUDENT;
```

ID	FIRST_NAME	AGE	COURSE
101	ARUN	20	CSE
102	BHAVESH	21	ISE
103	CHAITANYA	22	ECE
104	DEEPIKA	23	MECH
105	DHANUSH	24	DS
106	EKTA	25	AI
107	GAURAV	26	ARCH
108	HARSHITA	27	CHEMICAL
109	ISHAAN	28	CIVIL
110	JANU	29	EEE
110	JANU	29	EEE
111	PRANAV	NULL	NULL

12 rows in set (0.00 sec)

## HOW TO UPDATE SINGLE COLUMN IN THE RECORD?

```
UPDATE STUDENT SET FIRST_NAME = 'RISHI' WHERE ID = 108;
```

Query OK, 1 row affected (0.01 sec)

## FSD Training Program

Rows matched: 1 Changed: 1 Warnings: 0

```
SELECT * FROM STUDENT;
```

ID	FIRST_NAME	AGE	COURSE
101	ARUN	20	CSE
102	BHAVESH	21	ISE
103	CHAITANYA	22	ECE
104	DEEPIKA	23	MECH
105	DHANUSH	24	DS
106	EKTA	25	AI
107	GAURAV	26	ARCH
108	RISHI	27	CHEMICAL
109	ISHAAN	28	CIVIL
110	JANU	29	EEE
110	JANU	29	EEE
111	PRANAV	NULL	NULL

12 rows in set (0.00 sec)

### HOW TO UPDATE MULTIPLE COLUMNS IN THE RECORD?

```
UPDATE STUDENT SET ID = 112, FIRST_NAME = 'RAJAT', AGE = 29, COURSE = 'AUTOMOBILE' WHERE ID = 105;
```

Query OK, 1 row affected (0.01 sec)

Rows matched: 1 Changed: 1 Warnings: 0

```
SELECT * FROM STUDENT;
```

ID	FIRST_NAME	AGE	COURSE
101	ARUN	20	CSE
102	BHAVESH	21	ISE
103	CHAITANYA	22	ECE
104	DEEPIKA	23	MECH
112	RAJAT	29	AUTOMOBILE
106	EKTA	25	AI
107	GAURAV	26	ARCH
108	RISHI	27	CHEMICAL
109	ISHAAN	28	CIVIL
110	JANU	29	EEE
110	JANU	29	EEE
111	PRANAV	NULL	NULL

12 rows in set (0.00 sec)

## WHAT IS `NULL` IN SQL?

NULL is a special marker in SQL that represents the absence of a value or a undefined value in a database.

**Note:** `NULL` is case insensitive

## HOW TO USE `IS NULL`?

IS NULL is a condition used to check if a particular column in a database table has a NULL value.

```
UPDATE STUDENT SET AGE = 30 WHERE AGE IS NULL;
```

Query OK, 1 row affected (0.01 sec)

Rows matched: 1 Changed: 1 Warnings: 0

```
SELECT * FROM STUDENT;
```

ID	FIRST_NAME	AGE	COURSE
101	ARUN	20	CSE
102	BHAVESH	21	ISE
103	CHAITANYA	22	ECE
104	DEEPIKA	23	MECH
112	RAJAT	29	AUTOMOBILE
106	EKTA	25	AI
107	GAURAV	26	ARCH
108	RISHI	27	CHEMICAL
109	ISHAAN	28	CIVIL
110	JANU	29	EEE
110	JANU	29	EEE
111	PRANAV	30	NULL

12 rows in set (0.00 sec)

## HOW TO USE `IS NOT NULL`?

The IS NOT NULL condition is used to filter rows where a particular column does not contain a NULL value. It is the opposite of the IS NULL condition.

```
UPDATE STUDENT SET AGE = 20 WHERE FIRST_NAME IS NOT NULL;
```

```
Query OK, 11 rows affected (0.01 sec)
```

```
Rows matched: 12  Changed: 11  Warnings: 0
```

```
SELECT * FROM STUDENT;
```

ID	FIRST_NAME	AGE	COURSE
101	ARUN	20	CSE
102	BHAVESH	20	ISE
103	CHAITANYA	20	ECE
104	DEEPIKA	20	MECH
112	RAJAT	20	AUTOMOBILE
106	EKTA	20	AI
107	GAURAV	20	ARCH
108	RISHI	20	CHEMICAL
109	ISHAAN	20	CIVIL
110	JANU	20	EEE
110	JANU	20	EEE
111	PRANAV	20	NULL

```
12 rows in set (0.00 sec)
```

## HOW TO DELETE ALL THE RECORDS FROM A TABLE?

DELETE FROM STUDENT;

Query OK, 12 rows affected (0.01 sec)

## HOW TO INSERT RECORDS USING A SINGLE STATEMENT?

INSERT INTO STUDENT VALUES

```
(101, 'ARUN', 20, 'CSE'),
(102, 'BHAVESH', 21, 'ISE'),
(103, 'CHAITANYA', 22, 'ECE'),
(104, 'DEEPIKA', 23, 'MECH'),
(105, 'DHANUSH', 24, 'DS'),
(106, 'EKTA', 25, 'AI'),
(107, 'GAURAV', 26, 'ARCH'),
(108, 'HARSHITA', 27, 'CHEMICAL'),
(109, 'ISHAAN', 28, 'CIVIL'),
(110, 'JANU', 29, 'EEE');
```

SELECT \* FROM STUDENT;

```
+-----+-----+-----+-----+
| ID    | FIRST_NAME | AGE  | COURSE |
+-----+-----+-----+-----+
| 101   | ARUN       | 20   | CSE    |
| 102   | BHAVESH    | 21   | ISE    |
| 103   | CHAITANYA  | 22   | ECE    |
| 104   | DEEPIKA    | 23   | MECH   |
| 105   | DHANUSH    | 24   | DS     |
| 106   | EKTA       | 25   | AI     |
```

## FSD Training Program

	107	GAURAV		26	ARCH	
	108	HARSHITA		27	CHEMICAL	
	109	ISHAAN		28	CIVIL	
	110	JANU		29	EEE	

+-----+-----+-----+-----+

10 rows in set (0.00 sec)

HOW WOULD YOU UPDATE THE FIRST NAME COLUMN IN THE STUDENT TABLE FOR ALL RECORDS WHERE THE ID IS GREATER THAN 104, SETTING THE FIRST NAME TO 'ANANYA'?

UPDATE STUDENT SET FIRST\_NAME = 'ANANYA' WHERE ID > 104;

Query OK, 6 rows affected (0.01 sec)

Rows matched: 6 Changed: 6 Warnings: 0

SELECT \* FROM STUDENT;

+-----+-----+-----+-----+

	ID		FIRST_NAME		AGE		COURSE	
--	----	--	------------	--	-----	--	--------	--

+-----+-----+-----+-----+

	101		ARUN		20		CSE	
--	-----	--	------	--	----	--	-----	--

	102		BHAVESH		21		ISE	
--	-----	--	---------	--	----	--	-----	--

	103		CHAITANYA		22		ECE	
--	-----	--	-----------	--	----	--	-----	--

	104		DEEPIKA		23		MECH	
--	-----	--	---------	--	----	--	------	--

	105		ANANYA		24		DS	
--	-----	--	--------	--	----	--	----	--

	106		ANANYA		25		AI	
--	-----	--	--------	--	----	--	----	--

	107		ANANYA		26		ARCH	
--	-----	--	--------	--	----	--	------	--

	108		ANANYA		27		CHEMICAL	
--	-----	--	--------	--	----	--	----------	--



## FSD Training Program

	109		ANANYA		28		CIVIL	
	110		ANANYA		29		EEE	

+-----+-----+-----+-----+

10 rows in set (0.00 sec)

## HOW WOULD YOU UPDATE MULTIPLE COLUMNS?

UPDATE STUDENT SET AGE = 22, ID = 10 WHERE ID <= 107;

Query OK, 7 rows affected (0.00 sec)

Rows matched: 7 Changed: 7 Warnings: 0

SELECT \* FROM STUDENT;

+-----+-----+-----+-----+
ID   FIRST_NAME   AGE   COURSE
+-----+-----+-----+-----+
10   ARUN   22   CSE
10   BHAVESH   22   ISE
10   CHAITANYA   22   ECE
10   DEEPIKA   22   MECH
10   ANANYA   22   DS
10   ANANYA   22   AI
10   ANANYA   22   ARCH
108   ANANYA   27   CHEMICAL
109   ANANYA   28   CIVIL
110   ANANYA   29   EEE
+-----+-----+-----+-----+

10 rows in set (0.00 sec)

## HOW WOULD YOU UPDATE ALL THE COLUMNS?

```
UPDATE STUDENT SET AGE = 42, ID = 15;
```

Query OK, 10 rows affected (0.00 sec)

Rows matched: 10 Changed: 10 Warnings: 0

```
SELECT * FROM STUDENT;
```

ID	FIRST_NAME	AGE	COURSE
15	ARUN	42	CSE
15	BHAVESH	42	ISE
15	CHAITANYA	42	ECE
15	DEEPIKA	42	MECH
15	ANANYA	42	DS
15	ANANYA	42	AI
15	ANANYA	42	ARCH
15	ANANYA	42	CHEMICAL
15	ANANYA	42	CIVIL
15	ANANYA	42	EEE

10 rows in set (0.00 sec)

```
DELETE FROM STUDENT;
```

Query OK, 10 rows affected (0.01 sec)

## HOW WOULD YOU EXECUTE MULTIPLE STATEMENTS IN THE SQL WORKBENCH?

### 1. Write Your SQL Statements:

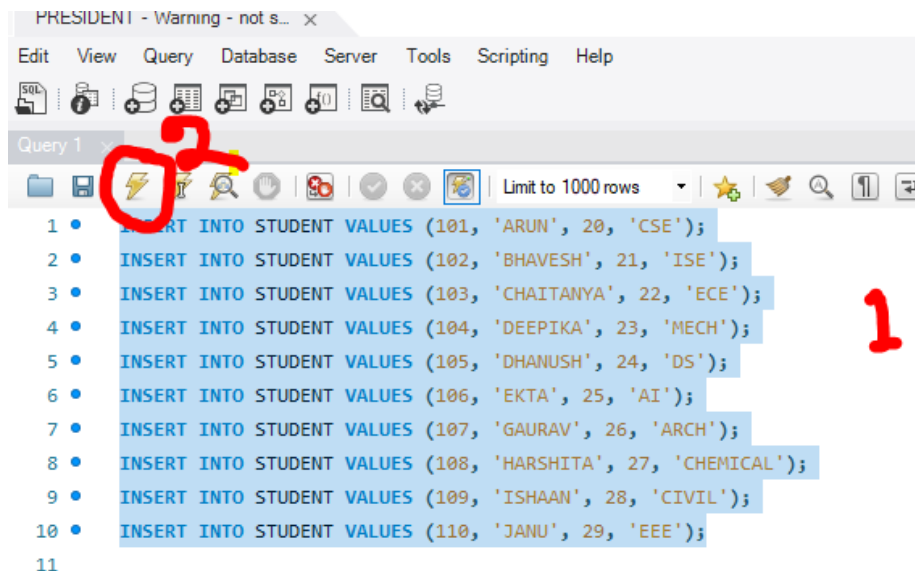
Open SQL Workbench and write the SQL statements you want to execute. Separate each statement with a semicolon (;).

### 2. Highlight the Statements:

Highlight all the SQL statements you want to execute.

### 3. Execute the Statements:

Execute the highlighted statements by either clicking on the "Execute" button (flash symbol), or pressing the appropriate shortcut (e.g., F5), or selecting the "Execute SQL" option from the menu.



```
INSERT INTO STUDENT VALUES (101, 'ARUN', 20, 'CSE');
INSERT INTO STUDENT VALUES (102, 'BHAVESH', 21, 'ISE');
INSERT INTO STUDENT VALUES (103, 'CHAITANYA', 22, 'ECE');
INSERT INTO STUDENT VALUES (104, 'DEEPIKA', 23, 'MECH');
INSERT INTO STUDENT VALUES (105, 'DHANUSH', 24, 'DS');
INSERT INTO STUDENT VALUES (106, 'EKTA', 25, 'AI');
INSERT INTO STUDENT VALUES (107, 'GAURAV', 26, 'ARCH');
INSERT INTO STUDENT VALUES (108, 'HARSHITA', 27,
'CHEMICAL');
```

## FSD Training Program

```
INSERT INTO STUDENT VALUES (109, 'ISHAAN', 28, 'CIVIL');
```

```
INSERT INTO STUDENT VALUES (110, 'JANU', 29, 'EEE');
```

```
SELECT * FROM STUDENT;
```

```
+-----+-----+-----+-----+
| ID    | FIRST_NAME | AGE  | COURSE |
+-----+-----+-----+-----+
| 101   | ARUN       | 20   | CSE    |
| 102   | BHAVESH    | 21   | ISE    |
| 103   | CHAITANYA  | 22   | ECE    |
| 104   | DEEPIKA    | 23   | MECH   |
| 105   | DHANUSH    | 24   | DS     |
| 106   | EKTA       | 25   | AI     |
| 107   | GAURAV     | 26   | ARCH   |
| 108   | HARSHITA   | 27   | CHEMICAL |
| 109   | ISHAAN     | 28   | CIVIL  |
| 110   | JANU       | 29   | EEE    |
+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

HOW WOULD YOU HOW YOU WOULD USE AN SQL DELETE STATEMENT TO REMOVE A SPECIFIC STUDENT RECORD WITH THE ID OF 6 FROM THE STUDENT TABLE?

```
DELETE FROM STUDENT WHERE ID = 6;
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
DELETE FROM STUDENT WHERE FIRST_NAME = 'ISHAAN';
```

Query OK, 1 row affected (0.00 sec)

SELECT \* FROM STUDENT;

ID	FIRST_NAME	AGE	COURSE
101	ARUN	20	CSE
102	BHAVESH	21	ISE
103	CHAITANYA	22	ECE
104	DEEPIKA	23	MECH
105	DHANUSH	24	DS
106	EKTA	25	AI
107	GAURAV	26	ARCH
108	HARSHITA	27	CHEMICAL
110	JANU	29	EEE

9 rows in set (0.00 sec)

DELETE FROM STUDENT;

Query OK, 9 rows affected (0.01 sec)

SELECT \* FROM STUDENT;

Empty set (0.00 sec)

## HOW WOULD YOU REMOVE A TABLE FROM THE DATABASE?

DROP TABLE STUDENT;

Query OK, 0 rows affected (0.02 sec)

```
CREATE TABLE EMPLOYEE (ID INTEGER, FIRST_NAME VARCHAR(90),  
LAST_NAME VARCHAR(90), AGE INTEGER, SALARY INTEGER, EMAIL  
VARCHAR(90));
```

Query OK, 0 rows affected (0.03 sec)

```
INSERT INTO EMPLOYEE VALUES(1, 'ARUN', 'PATEL', 22, 40000,  
'ARUN@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(2, 'BHAVESH', 'SHARMA', 24,  
30000, 'BHAVESH@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(3, 'CHAITANYA', 'SINGH', 23,  
50000, 'CHAITANYA@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(4, 'DEEPIKA', 'GUPTA', 26,  
55000, 'DEEPIKA@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(5, 'DHANUSH', 'KUMAR', 25,  
20000, 'DHANUSH@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(6, 'EKTA', 'YADAV', 28, 35000,  
'YADAV@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(7, 'GAURAV', 'RAO', 21, 60000,  
'GAURAV@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(8, 'HARSHITA', 'REDDY', 29,  
56000, 'HARSHITA@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(9, 'ISHAAN', 'REDDY', 32,  
70000, 'ISHAAN@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(10, 'JANU', 'MUKHERJEE', 30,  
53000, 'JANU@GCOMPANY.IN');
```

## FSD Training Program

**SELECT \* FROM EMPLOYEE;**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
3	CHAITANYA	SINGH	23	50000	CHAITANYA@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
5	DHANUSH	KUMAR	25	20000	DHANUSH@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN

10 rows in set (0.00 sec)

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE WITH THE ID OF 5 FROM THE EMPLOYEE TABLE?**

**SELECT \* FROM EMPLOYEE WHERE ID = 5;**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
5	DHANUSH	KUMAR	25	20000	DHANUSH@GCOMPANY.IN

1 row in set (0.00 sec)

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE WITH THE ID GREATER THAN 5 FROM THE EMPLOYEE TABLE?**

**SELECT \* FROM EMPLOYEE WHERE ID > 5;**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN

5 rows in set (0.00 sec)

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE WHERE AGE RANGE OF 22 TO 28 FROM THE EMPLOYEE TABLE?**

**SELECT \* FROM EMPLOYEE WHERE AGE BETWEEN 22 AND 28;**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
3	CHAITANYA	SINGH	23	50000	CHAITANYA@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
5	DHANUSH	KUMAR	25	20000	DHANUSH@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN

6 rows in set (0.00 sec)



**WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE WHERE AGE NOT IN THE RANGE OF 22 TO 28 FROM THE EMPLOYEE TABLE?**

**SELECT \* FROM EMPLOYEE WHERE AGE NOT BETWEEN 22 AND 28;**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN

4 rows in set (0.00 sec)

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE RETRIEVE DETAILS FOR EMPLOYEES WHOSE SALARIES MATCH SPECIFIC VALUES FROM THE EMPLOYEE TABLE?**

**SELECT \* FROM EMPLOYEE WHERE SALARY IN (40000, 55000, 70000);**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN

3 rows in set (0.00 sec)

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE RETRIEVE DETAILS FOR EMPLOYEES WHOSE SALARIES DOESNT MATCH SPECIFIC VALUES FROM THE EMPLOYEE TABLE?**

**SELECT \* FROM EMPLOYEE WHERE SALARY NOT IN (40000, 55000, 70000);**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
3	CHAITANYA	SINGH	23	50000	CHAITANYA@GCOMPANY.IN
5	DHANUSH	KUMAR	25	20000	DHANUSH@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN

7 rows in set (0.00 sec)

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE WHERE NAMES OF ALL EMPLOYEES WHOSE FIRST NAME INCLUDES THE LETTER "R" FROM THE EMPLOYEE TABLE?**

**SELECT \* FROM EMPLOYEE WHERE FIRST\_NAME LIKE '%R%';**

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN

3 rows in set (0.00 sec)

WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE WHERE NAMES OF ALL EMPLOYEES WHOSE FIRST NAME ENDING WITH THE LETTER "R" FROM THE EMPLOYEE TABLE?

SELECT \* FROM EMPLOYEE WHERE FIRST\_NAME LIKE '%A';

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 3     | CHAITANYA  | SINGH     | 23   | 50000  | CHAITANYA@GCOMPANY.IN |
| 4     | DEEPIKA   | GUPTA     | 26   | 55000  | DEEPIKA@GCOMPANY.IN  |
| 6     | EKTA      | YADAV     | 28   | 35000  | YADAV@GCOMPANY.IN    |
| 8     | HARSHITA  | REDDY     | 29   | 56000  | HARSHITA@GCOMPANY.IN |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

WRITE AN SQL QUERY THAT RETRIEVES RECORDS OF THE EMPLOYEE TABLE WHERE NAMES OF ALL EMPLOYEES WHOSE FIRST NAME STARTING WITH THE LETTER "R" FROM THE EMPLOYEE TABLE?

SELECT \* FROM EMPLOYEE WHERE FIRST\_NAME LIKE 'A%';

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 1     | ARUN       | PATEL     | 22   | 40000  | ARUN@GCOMPANY.IN |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS WHERE FIRST NAMES AND AGES OF ALL EMPLOYEES FROM THE EMPLOYEE TABLE?**

**SELECT FIRST\_NAME, AGE FROM EMPLOYEE;**

+-----+-----+		
FIRST_NAME	AGE	
+-----+-----+		
ARUN	22	
BHAVESH	24	
CHAITANYA	23	
DEEPIKA	26	
DHANUSH	25	
EKTA	28	
GAURAV	21	
HARSHITA	29	
ISHAAN	32	
JANU	30	
+-----+-----+		

**10 rows in set (0.00 sec)**

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS WHERE FIRST NAMES, AGE AND EMAIL OF ALL EMPLOYEES FROM THE EMPLOYEE TABLE?**

**SELECT FIRST\_NAME, AGE, EMAIL FROM EMPLOYEE;**

```
+-----+-----+-----+
| FIRST_NAME | AGE | EMAIL |
+-----+-----+-----+
| ARUN       | 22  | ARUN@GCOMPANY.IN |
| BHAVESH    | 24  | BHAVESH@GCOMPANY.IN |
| CHAITANYA  | 23  | CHAITANYA@GCOMPANY.IN |
| DEEPIKA    | 26  | DEEPIKA@GCOMPANY.IN |
| DHANUSH    | 25  | DHANUSH@GCOMPANY.IN |
| EKTA       | 28  | YADAV@GCOMPANY.IN |
| GAURAV     | 21  | GAURAV@GCOMPANY.IN |
| HARSHITA   | 29  | HARSHITA@GCOMPANY.IN |
| ISHAAN     | 32  | ISHAAN@GCOMPANY.IN |
| JANU       | 30  | JANU@GCOMPANY.IN |
+-----+-----+-----+
10 rows in set (0.00 sec)
```

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS USING ALIAS NAMES FOR THE COLUMNS USING `AS` KEYWORD FROM THE EMPLOYEE TABLE?**

**SELECT FIRST\_NAME AS NAME, AGE AS EMPLOYEE\_AGE, LAST\_NAME  
FROM EMPLOYEE;**

+-----+-----+-----+			
NAME	EMPLOYEE_AGE	LAST_NAME	
+-----+-----+-----+			
ARUN	22	PATEL	
BHAVESH	24	SHARMA	
CHAITANYA	23	SINGH	
DEEPIKA	26	GUPTA	
DHANUSH	25	KUMAR	
EKTA	28	YADAV	
GAURAV	21	RAO	
HARSHITA	29	REDDY	
ISHAAN	32	REDDY	
JANU	30	MUKHERJEE	
+-----+-----+-----+			

**10 rows in set (0.00 sec)**

**WRITE AN SQL QUERY THAT RETRIEVES RECORDS USING ALIAS NAMES FOR THE COLUMNS WITHOUT USING `AS` KEYWORD FROM THE EMPLOYEE TABLE?**

```
SELECT FIRST_NAME NAME, AGE EMPLOYEE_AGE, LAST_NAME FROM  
EMPLOYEE;
```

NAME	EMPLOYEE_AGE	LAST_NAME
ARUN	22	PATEL
BHAVESH	24	SHARMA
CHAITANYA	23	SINGH
DEEPIKA	26	GUPTA
DHANUSH	25	KUMAR
EKTA	28	YADAV
GAURAV	21	RAO
HARSHITA	29	REDDY
ISHAAN	32	REDDY
JANU	30	MUKHERJEE

10 rows in set (0.00 sec)

**WRITE AN SQL QUERY THAT RETRIEVES THE TOTAL NUMBER OF EMPLOYEES IN THE COMPANY?**

The COUNT(\*) function in SQL is used to count the number of rows in a table or the result set of a query. It can be used in various ways to analyze and retrieve information from your data.

## FSD Training Program

Here's a breakdown of what COUNT(\*) does:

**Counts all rows:** The asterisk (\*) indicates that all columns in every row should be counted, regardless of their value (including null values).

**Returns an integer:** The function returns a single integer value representing the total number of rows counted.

**Used in SELECT statements:** COUNT(\*) is typically used within a SELECT statement, often in conjunction with other functions like WHERE clauses to filter the data before counting.

### EXAMPLE: 1

```
SELECT COUNT(*) FROM EMPLOYEE;
```

```
+-----+
```

```
| COUNT(*) |
```

```
+-----+
```

```
|      10 |
```

```
+-----+
```

```
1 row in set (0.01 sec)
```

### EXAMPLE: 2

```
SELECT COUNT(*) AS "RECORDS COUNT" FROM EMPLOYEE;
```

```
+-----+
```

```
| RECORDS COUNT |
```

```
+-----+
```

```
|           10 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```



**EXAMPLE: 3**

```
SELECT COUNT(*) "RECORDS COUNT" FROM EMPLOYEE;
```

```
+-----+
```

```
| RECORDS COUNT |
```

```
+-----+
```

```
|          10 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**RETRIEVE THE COUNT OF EMPLOYEE RECORDS WHOSE AGE COLUMN HAVING A VALUE EXCEPT `NULL`?**

```
SELECT COUNT(AGE) "AGE COLUMN COUNT" FROM EMPLOYEE;
```

```
+-----+
```

```
| AGE COLUMN COUNT |
```

```
+-----+
```

```
|          10 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**RETRIEVE THE COUNT OF EMPLOYEE RECORDS WHOSE LAST NAME COLUMN HAVING A VALUE EXCEPT `NULL`?**

```
SELECT COUNT(LAST_NAME) "RECORDS COUNT" FROM EMPLOYEE;
```

```
+-----+
```

```
| RECORDS COUNT |
```

```
+-----+
```

```
|          10 |
```

```
+-----+
```

1 row in set (0.01 sec)

## RETRIEVE MAXIMUM AGE FROM THE EMPLOYEE TABLE?

### EXAMPLE: 1

```
SELECT MAX(AGE) FROM EMPLOYEE;
```

```
+-----+
```

```
| MAX(AGE) |
```

```
+-----+
```

```
|      32 |
```

```
+-----+
```

1 row in set (0.00 sec)

### EXAMPLE: 2

```
SELECT MAX(AGE) AS "MAX AGE" FROM EMPLOYEE;
```

```
+-----+
```

```
| MAX AGE |
```

```
+-----+
```

```
|      32 |
```

```
+-----+
```

1 row in set (0.00 sec)

## RETRIEVE MINIMUM SALARY FROM THE EMPLOYEE TABLE?

### EXAMPLE: 1

```
SELECT MIN(SALARY) FROM EMPLOYEE;
```

```
+-----+
```

```
| MIN(SALARY) |
```

```
+-----+
```

```
|      20000 |
```

## FSD Training Program

+-----+

1 row in set (0.00 sec)

### EXAMPLE: 2

```
SELECT MIN(SALARY) MIN_SAL FROM EMPLOYEE;
```

+-----+

MIN_SAL
20000

+-----+

+-----+

1 row in set (0.00 sec)

### EXAMPLE: 3

```
SELECT MIN(SALARY) "MIN SAL" FROM EMPLOYEE;
```

+-----+

MIN SAL
20000

+-----+

+-----+

1 row in set (0.00 sec)

## RETRIEVE AVERAGE SALARY FROM THE EMPLOYEE TABLE?

### EXAMPLE: 1

```
SELECT AVG(SALARY) FROM EMPLOYEE;
```

+-----+

AVG(SALARY)
46900.0000

+-----+

+-----+

1 row in set (0.00 sec)

**EXAMPLE: 2**

```
SELECT AVG(SALARY) "AVG SALARY" FROM EMPLOYEE;
```

```
+-----+
```

```
|  AVG SALARY  |
```

```
+-----+
```

```
| 46900.0000 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**EXAMPLE: 3**

```
SELECT AVG(AGE) "AVG AGE" FROM EMPLOYEE;
```

```
+-----+
```

```
|  AVG AGE  |
```

```
+-----+
```

```
| 26.0000 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**RETRIEVE THE MINIMUM ASCII VALUE AMONG THE VALUES OF  
FIRST NAME COLUMN FROM THE EMPLOYEE TABLE?**

```
SELECT MIN(FIRST_NAME) FROM EMPLOYEE;
```

```
+-----+
```

```
| MIN(FIRST_NAME) |
```

```
+-----+
```

```
|  ARUN           |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

## RETRIEVE THE MAX ASCII VALUE AMONG THE VALUES OF FIRST NAME COLUMN FROM THE EMPLOYEE TABLE?

```
SELECT MAX(FIRST_NAME) FROM EMPLOYEE;
```

```
+-----+
```

```
| MAX(FIRST_NAME) |
```

```
+-----+
```

```
| JANU          |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

## DEMONSTRATE ORDER BY

```
SELECT * FROM EMPLOYEE ORDER BY FIRST_NAME;
```

- ORDER BY in MySQL is like telling the database how you want your results to be arranged or sorted when you retrieve them from a table.

- It is commonly used in conjunction with the SELECT statement.

- default sorting is ascending order.

## ORDER EMPLOYEE RECORDS USING ORDER BY IN THE ASCENDING ORDER BY CONSIDERING FIRST NAME COLUMN?

```
SELECT * FROM EMPLOYEE ORDER BY FIRST_NAME;
```

```
+-----+-----+-----+-----+-----+-----+
```

```
| ID | FIRST_NAME | LAST_NAME | AGE | SALARY | EMAIL |
```

```
+-----+-----+-----+-----+-----+-----+
```

```
| 1 | ARUN | PATEL | 22 | 40000 | ARUN@GCOMPANY.IN |
```

```
| 2 | BHAVESH | SHARMA | 24 | 30000 | BHAVESH@GCOMPANY.IN |
```

```
| 3 | CHAITANYA | SINGH | 23 | 50000 | CHAITANYA@GCOMPANY.IN |
```

```
| 4 | DEEPIKA | GUPTA | 26 | 55000 | DEEPIKA@GCOMPANY.IN |
```

```
| 5 | DHANUSH | KUMAR | 25 | 20000 | DHANUSH@GCOMPANY.IN |
```

```
| 6 | EKTA | YADAV | 28 | 35000 | YADAV@GCOMPANY.IN |
```

## FSD Training Program

7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN

+-----+-----+-----+-----+-----+-----+

10 rows in set (0.00 sec)

### ORDER EMPLOYEE RECORDS USING ORDER BY IN THE ASCENDING ORDER BY CONSIDERING FIRST NAME COLUMN USING AS KEYWORD?

SELECT \* FROM EMPLOYEE ORDER BY FIRST\_NAME ASC;

+-----+-----+-----+-----+-----+-----+

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
----	------------	-----------	-----	--------	-------

+-----+-----+-----+-----+-----+-----+

1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
3	CHAITANYA	SINGH	23	50000	CHAITANYA@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
5	DHANUSH	KUMAR	25	20000	DHANUSH@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN

+-----+-----+-----+-----+-----+-----+

10 rows in set (0.00 sec)

### ORDER EMPLOYEE RECORDS USING ORDER BY IN THE DESCENDING ORDER BY CONSIDERING FIRST NAME COLUMN?

SELECT \* FROM EMPLOYEE ORDER BY FIRST\_NAME DESC;

+-----+-----+-----+-----+-----+-----+

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
----	------------	-----------	-----	--------	-------

+-----+-----+-----+-----+-----+-----+

## FSD Training Program

	10		JANU		MUKHERJEE		30		53000		JANU@GCOMPANY.IN	
	9		ISHAAN		REDDY		32		70000		ISHAAN@GCOMPANY.IN	
	8		HARSHITA		REDDY		29		56000		HARSHITA@GCOMPANY.IN	
	7		GAURAV		RAO		21		60000		GAURAV@GCOMPANY.IN	
	6		EKTA		YADAV		28		35000		YADAV@GCOMPANY.IN	
	5		DHANUSH		KUMAR		25		20000		DHANUSH@GCOMPANY.IN	
	4		DEEPIKA		GUPTA		26		55000		DEEPIKA@GCOMPANY.IN	
	3		CHAITANYA		SINGH		23		50000		CHAITANYA@GCOMPANY.IN	
	2		BHAVESH		SHARMA		24		30000		BHAVESH@GCOMPANY.IN	
	1		ARUN		PATEL		22		40000		ARUN@GCOMPANY.IN	

+-----+-----+-----+-----+-----+-----+-----+

10 rows in set (0.00 sec)

## ORDER EMPLOYEE RECORDS USING ORDER BY IN THE ASCENDING ORDER BY CONSIDERING AGE COLUMN?

SELECT \* FROM EMPLOYEE ORDER BY AGE;

+-----+-----+-----+-----+-----+-----+-----+

	ID		FIRST_NAME		LAST_NAME		AGE		SALARY		EMAIL	
	7		GAURAV		RAO		21		60000		GAURAV@GCOMPANY.IN	
	1		ARUN		PATEL		22		40000		ARUN@GCOMPANY.IN	
	3		CHAITANYA		SINGH		23		50000		CHAITANYA@GCOMPANY.IN	
	2		BHAVESH		SHARMA		24		30000		BHAVESH@GCOMPANY.IN	
	5		DHANUSH		KUMAR		25		20000		DHANUSH@GCOMPANY.IN	
	4		DEEPIKA		GUPTA		26		55000		DEEPIKA@GCOMPANY.IN	
	6		EKTA		YADAV		28		35000		YADAV@GCOMPANY.IN	
	8		HARSHITA		REDDY		29		56000		HARSHITA@GCOMPANY.IN	
	10		JANU		MUKHERJEE		30		53000		JANU@GCOMPANY.IN	
	9		ISHAAN		REDDY		32		70000		ISHAAN@GCOMPANY.IN	

+-----+-----+-----+-----+-----+-----+-----+

10 rows in set (0.00 sec)

## ORDER EMPLOYEE RECORDS USING ORDER BY IN THE ASCENDING ORDER BY CONSIDERING SALARY COLUMN?

```
SELECT * FROM EMPLOYEE ORDER BY SALARY;
```

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 5     | DHANUSH    | KUMAR     | 25   | 20000  | DHANUSH@GCOMPANY.IN                |
| 2     | BHAVESH    | SHARMA    | 24   | 30000  | BHAVESH@GCOMPANY.IN                |
| 6     | EKTA       | YADAV     | 28   | 35000  | YADAV@GCOMPANY.IN                  |
| 1     | ARUN       | PATEL     | 22   | 40000  | ARUN@GCOMPANY.IN                   |
| 3     | CHAITANYA  | SINGH     | 23   | 50000  | CHAITANYA@GCOMPANY.IN              |
| 10    | JANU       | MUKHERJEE | 30   | 53000  | JANU@GCOMPANY.IN                   |
| 4     | DEEPIKA    | GUPTA     | 26   | 55000  | DEEPIKA@GCOMPANY.IN                |
| 8     | HARSHITA   | REDDY     | 29   | 56000  | HARSHITA@GCOMPANY.IN               |
| 7     | GAURAV     | RAO       | 21   | 60000  | GAURAV@GCOMPANY.IN                 |
| 9     | ISHAAN     | REDDY     | 32   | 70000  | ISHAAN@GCOMPANY.IN                 |
+-----+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```

## ORDER EMPLOYEE RECORDS USING ORDER BY IN THE ASCENDING ORDER BY CONSIDERING AGE AND SALARY COLUMNS?

**Note:** The first preference would be for the first column. If there are rows with the same value in column1, those rows will then be sorted by the values in column2 in ascending order. The default sorting is ascending order.

```
SELECT * FROM EMPLOYEE ORDER BY AGE, SALARY;
```

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 7     | GAURAV     | RAO       | 21   | 60000  | GAURAV@GCOMPANY.IN                |
+-----+-----+-----+-----+-----+-----+
```



## FSD Training Program

	1		ARUN		PATEL		22		40000		ARUN@GCOMPANY.IN	
	3		CHAITANYA		SINGH		23		50000		CHAITANYA@GCOMPANY.IN	
	2		BHAVESH		SHARMA		24		30000		BHAVESH@GCOMPANY.IN	
	5		DHANUSH		KUMAR		25		20000		DHANUSH@GCOMPANY.IN	
	4		DEEPIKA		GUPTA		26		55000		DEEPIKA@GCOMPANY.IN	
	6		EKTA		YADAV		28		35000		YADAV@GCOMPANY.IN	
	8		HARSHITA		REDDY		29		56000		HARSHITA@GCOMPANY.IN	
	10		JANU		MUKHERJEE		30		53000		JANU@GCOMPANY.IN	
	9		ISHAAN		REDDY		32		70000		ISHAAN@GCOMPANY.IN	
	11		ARUL		PATEL		35		40000		ARUL@GCOMPANY.IN	
	12		ADITI		PATEL		35		60000		ADITI@GCOMPANY.IN	

+-----+-----+-----+-----+-----+-----+-----+

12 rows in set (0.00 sec)

```
INSERT INTO EMPLOYEE VALUES(11, 'ARUL', 'PATEL', 35, 40000,
'ARUL@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(12, 'ADITI', 'PATEL', 35,
60000, 'ADITI@GCOMPANY.IN');
```

## ORDER EMPLOYEE RECORDS USING ORDER BY AND AGE IN ASCENDING ORDER AND SALARY IN DESC ORDER?

```
SELECT * FROM EMPLOYEE ORDER BY AGE ASC, SALARY DESC;
```

**Note:** First considers the first column and sorts in the specified order, if two values of the first column are the same then it considers the second column and sorts in the specified order.

+-----+-----+-----+-----+-----+-----+-----+						
ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL	
+-----+-----+-----+-----+-----+-----+-----+						
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN	
1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN	
3	CHAITANYA	SINGH	23	50000	CHAITANYA@GCOMPANY.IN	

## FSD Training Program

	2		BHAVESH		SHARMA		24		30000		BHAVESH@GCOMPANY.IN	
	5		DHANUSH		KUMAR		25		20000		DHANUSH@GCOMPANY.IN	
	4		DEEPIKA		GUPTA		26		55000		DEEPIKA@GCOMPANY.IN	
	6		EKTA		YADAV		28		35000		YADAV@GCOMPANY.IN	
	8		HARSHITA		REDDY		29		56000		HARSHITA@GCOMPANY.IN	
	10		JANU		MUKHERJEE		30		53000		JANU@GCOMPANY.IN	
	9		ISHAAN		REDDY		32		70000		ISHAAN@GCOMPANY.IN	
	12		ADITI		PATEL		35		60000		ADITI@GCOMPANY.IN	
	11		ARUL		PATEL		35		40000		ARUL@GCOMPANY.IN	

+-----+-----+-----+-----+-----+-----+-----+

12 rows in set (0.00 sec)

## WHAT IF TWO VALUES OF THE SAME COLUMN ARE SAME?

INSERT INTO EMPLOYEE VALUES(13, 'ARTI', ' PATEL', 35, 10000, ' ARTI@GCOMPANY.IN');

SELECT \* FROM EMPLOYEE ORDER BY AGE, SALARY;

Note: Then it considers the second column minimum value

EMPLOYEE INFORMATION						
ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL	
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN	
1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN	
3	CHAITANYA	SINGH	23	50000	CHAITANYA@GCOMPANY.IN	
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN	
5	DHANUSH	KUMAR	25	20000	DHANUSH@GCOMPANY.IN	
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN	
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN	
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN	
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN	
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN	
13	ARTI	PATEL	35	10000	ARTI@GCOMPANY.IN	
11	ARUL	PATEL	35	40000	ARUL@GCOMPANY.IN	

## FSD Training Program

	12		ADITI		PATEL		35		60000		ADITI@GCOMPANY.IN	
+-----+-----+-----+-----+-----+-----+												

**Note:** We get error for the below query because MAX(SALARY) is not a column in the employee table.

```
//SELECT FIRST_NAME, MAX(SALARY) FROM EMPLOYEE;//ERROR
```

**WHICH EMPLOYEE HAS THE HIGHEST SALARY, AND WHAT IS THEIR FIRST NAME?**

```
SELECT FIRST_NAME FROM EMPLOYEE WHERE SALARY = (SELECT  
MAX(SALARY) FROM EMPLOYEE);
```

```
+-----+
```

```
| FIRST_NAME |
```

```
+-----+
```

```
| ISHAAN    |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**WHICH EMPLOYEE HAS THE HIGHEST AGE, AND WHAT IS THEIR FIRST NAME?**

```
SELECT FIRST_NAME FROM EMPLOYEE WHERE AGE = (SELECT  
MAX(AGE) FROM EMPLOYEE);
```

```
+-----+
```

```
| FIRST_NAME |
```

```
+-----+
```

```
| ARUL       |
```

```
| ADITI      |
```

```
+-----+
```

**WHICH EMPLOYEE HAS THE LOWEST AGE, AND WHAT IS THEIR FIRST NAME?**

```
SELECT FIRST_NAME FROM EMPLOYEE WHERE AGE = (SELECT  
MIN(AGE) FROM EMPLOYEE);
```

```
+-----+
```

```
| FIRST_NAME |
```

```
+-----+
```

```
| GAURAV      |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**WHICH EMPLOYEE HAS LESS THAN AVERAGE SALARY, AND WHAT IS THEIR FIRST NAME?**

```
SELECT FIRST_NAME FROM EMPLOYEE WHERE SALARY < (SELECT  
AVG(SALARY) FROM EMPLOYEE);
```

```
+-----+
```

```
| FIRST_NAME |
```

```
+-----+
```

```
| ARUN        |
```

```
| BHAVESH     |
```

```
| DHANUSH     |
```

```
| EKTA        |
```

```
| ARUL        |
```

```
+-----+
```

```
5 rows in set (0.00 sec)
```

**WHAT IS THE MAXIMUM SALARY FROM THE EMPLOYEE TABLE?**

```
SELECT MAX(SALARY) FROM EMPLOYEE;
```

```
+-----+
```

```
| MAX(SALARY) |
```

```
+-----+
```

```
|      70000 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**WHAT IS THE SECOND MAXIMUM SALARY IN THE EMPLOYEE TABLE?**

```
SELECT MAX(SALARY) FROM EMPLOYEE WHERE SALARY < (SELECT  
MAX(SALARY) FROM EMPLOYEE);
```

```
+-----+
```

```
| MAX(SALARY) |
```

```
+-----+
```

```
|      60000 |
```

```
+-----+
```

```
1 row in set (0.00 sec)
```

**WHAT IS THE SECOND MINIMUM SALARY IN THE EMPLOYEE TABLE?**

```
SELECT MIN(SALARY) FROM EMPLOYEE WHERE SALARY > (SELECT  
MIN(SALARY) FROM EMPLOYEE);
```

```
+-----+
```

```
| MIN(SALARY) |
```

```
+-----+
```

```
|      30000 |
```

## FSD Training Program

+-----+

1 row in set (0.00 sec)

### WHICH EMPLOYEE HAS SECOND MINIMUM SALARY, AND WHAT IS THEIR FIRST NAME?

```
SELECT FIRST_NAME FROM EMPLOYEE WHERE SALARY = (SELECT  
MIN(SALARY) FROM EMPLOYEE WHERE SALARY > (SELECT  
MIN(SALARY) FROM EMPLOYEE));
```

+-----+

FIRST_NAME
------------

+-----+

BHAVESH
---------

+-----+

1 row in set (0.00 sec)

### WHICH EMPLOYEE HAS SECOND MAXIMUM SALARY, AND WHAT IS THEIR FIRST NAME?

```
SELECT FIRST_NAME FROM EMPLOYEE WHERE SALARY = (SELECT  
MAX(SALARY) FROM EMPLOYEE WHERE SALARY < (SELECT  
MAX(SALARY) FROM EMPLOYEE));
```

+-----+

FIRST_NAME
------------

+-----+

GAURAV
--------

ADITI
-------

+-----+

2 rows in set (0.00 sec)

## WHICH EMPLOYEE HAS SECOND MINIMUM SALARY, FETCH THEIR COMPLETE DETAILS?

```
SELECT * FROM EMPLOYEE WHERE SALARY = (SELECT MIN(SALARY)
FROM EMPLOYEE WHERE SALARY > (SELECT MIN(SALARY) FROM
EMPLOYEE));
```

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN

1 row in set (0.00 sec)

## WHAT IS THE EMPLOYEE ID, FIRST NAME, LAST NAME, AGE, SALARY, EMAIL, AND RANK BASED ON SALARY IN DESCENDING ORDER FOR ALL EMPLOYEES?

**Note:** the RANK() function is a window function that assigns a rank to each row within a partition of a result set. It is commonly used to assign a rank to rows based on the values in one or more columns.

- The RANK() function assigns a rank to each row based on its position in the ordered list within each partition.
- Ties are handled by assigning the same rank to all tied rows.

```
SELECT ID, FIRST_NAME, LAST_NAME, AGE, SALARY, EMAIL,
RANK() OVER (ORDER BY SALARY DESC) FROM EMPLOYEE;
```

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL	RANK() OVER (ORDER BY SALARY DESC)
9	ISHAAN	REDDY	32	70000	ISHAAN@GCOMPANY.IN	1
7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN	2
12	ADITI	PATEL	35	60000	ADITI@GCOMPANY.IN	2
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN	4
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN	5
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN	6

3	CHAITANYA	SINGH	23	50000	CHAITANYA@GCOMPANY.IN	7
1	ARUN	PATEL	22	40000	ARUN@GCOMPANY.IN	8
11	ARUL	PATEL	35	40000	ARUL@GCOMPANY.IN	8
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN	10
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN	11
5	DHANUSH	KUMAR	25	20000	DHANUSH@GCOMPANY.IN	12

12 rows in set (0.01 sec)

```
SELECT * FROM (SELECT ID, FIRST_NAME, LAST_NAME, AGE, SALARY, EMAIL, RANK() OVER(ORDER BY
SALARY DESC) RANKED_EMPLOYEES FROM EMPLOYEE) AS RANKED_EMPLOYEES WHERE RANKED_EMPLOYEES = 2;
```

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL	RANKED_EMPLOYEES
----	------------	-----------	-----	--------	-------	------------------

7	GAURAV	RAO	21	60000	GAURAV@GCOMPANY.IN	2
---	--------	-----	----	-------	--------------------	---

12	ADITI	PATEL	35	60000	ADITI@GCOMPANY.IN	2
----	-------	-------	----	-------	-------------------	---

```
2 rows in set (0.00 sec)
```

ID	FIRST NAME	LAST NAME	AGE	SALARY	EMAIL
----	------------	-----------	-----	--------	-------

1	ARUN	DATFI	22	10000	ARUN@COMPANY.IN
---	------	-------	----	-------	-----------------

2	BUAYESU	SHARMA	24	20000	BUAYESU@GCOMANY.TN
---	---------	--------	----	-------	--------------------

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 10

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

[illegible]

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	52
--	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	----

Table 1. *Continued*

9	HARSHITA	REDDY	ED	50000	HARSHITA REDDY ANANTH
---	----------	-------	----	-------	-----------------------

5	ISHAAN	REDDY	52	70000	ISHAAN@SECURIFY.IN
---	--------	-------	----	-------	--------------------



## FSD Training Program

	10		JANU		MUKHERJEE		30		53000		JANU@GCOMPANY.IN	
	11		ARUL		PATEL		35		40000		ARUL@GCOMPANY.IN	
	12		ADITI		PATEL		35		60000		ADITI@GCOMPANY.IN	

+-----+-----+-----+-----+-----+-----+-----+

12 rows in set (0.00 sec)

**DELETE FROM EMPLOYEE;**

**Query OK, 12 rows affected (0.01 sec)**

**INSERT INTO EMPLOYEE VALUES(1, 'ARUN', 'PATEL', 22, 40000, 'ARUN@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(2, 'BHAVESH', 'SHARMA', 24, 30000, 'BHAVESH@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(3, 'CHAITANYA', 'SINGH', 23, 50000, 'CHAITANYA@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(4, 'DEEPIKA', 'GUPTA', 26, 55000, 'DEEPIKA@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(5, 'DHANUSH', 'KUMAR', 25, 20000, 'DHANUSH@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(6, 'EKTA', 'YADAV', 28, 35000, 'YADAV@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(7, 'GAURAV', 'RAO', 21, 60000, 'GAURAV@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(8, 'HARSHITA', 'REDDY', 29, 56000, 'HARSHITA@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(9, 'ISHAAN', 'REDDY', 32, 70000, 'ISHAAN@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(10, 'JANU', 'MUKHERJEE', 30, 53000, 'JANU@GCOMPANY.IN');**

**INSERT INTO EMPLOYEE VALUES(1, 'ARUN', 'PATEL', 22, 40000, 'ARUN@GCOMPANY.IN');**

## FSD Training Program

```
INSERT INTO EMPLOYEE VALUES(2, 'BHAVESH', 'SHARMA', 24,
30000, 'BHAVESH@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(3, 'CHAITANYA', 'SINGH', 23,
50000, 'CHAITANYA@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(4, 'DEEPIKA', 'GUPTA', 26,
55000, 'DEEPIKA@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(5, 'DHANUSH', 'KUMAR', 25,
20000, 'DHANUSH@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(6, 'EKTA', 'YADAV', 28, 35000,
'YADAV@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(7, 'GAURAV', 'RAO', 21, 60000,
'GAURAV@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(8, 'HARSHITA', 'REDDY', 29,
56000, 'HARSHITA@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(9, 'ISHAAN', 'REDDY', 32,
70000, 'ISHAAN@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(10, 'JANU', 'MUKHERJEE', 30,
53000, 'JANU@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(1, 'ARUN', 'PATEL', 22, 40000,
'ARUN@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(2, 'BHAVESH', 'SHARMA', 24,
30000, 'BHAVESH@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(3, 'CHAITANYA', 'SINGH', 23,
50000, 'CHAITANYA@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(4, 'DEEPIKA', 'GUPTA', 26,
55000, 'DEEPIKA@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(5, 'DHANUSH', 'KUMAR', 25,
20000, 'DHANUSH@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(6, 'EKTA', 'YADAV', 28, 35000,
'YADAV@GCOMPANY.IN');

INSERT INTO EMPLOYEE VALUES(7, 'GAURAV', 'RAO', 21, 60000,
'GAURAV@GCOMPANY.IN');
```

## FSD Training Program

```
INSERT INTO EMPLOYEE VALUES(8, 'HARSHITA', 'REDDY', 29,  
56000, 'HARSHITA@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(9, 'ISHAAN', 'REDDY', 32,  
70000, 'ISHAAN@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(10, 'JANU', 'MUKHERJEE', 30,  
53000, 'JANU@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(1, 'ARUN', 'PATEL', 22, 40000,  
'ARUN@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(2, 'BHAVESH', 'SHARMA', 24,  
30000, 'BHAVESH@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(3, 'CHAITANYA', 'SINGH', 23,  
50000, 'CHAITANYA@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(4, 'DEEPIKA', 'GUPTA', 26,  
55000, 'DEEPIKA@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(5, 'DHANUSH', 'KUMAR', 25,  
20000, 'DHANUSH@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(6, 'EKTA', 'YADAV', 28, 35000,  
'YADAV@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(7, 'GAURAV', 'RAO', 21, 60000,  
'GAURAV@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(8, 'HARSHITA', 'REDDY', 29,  
56000, 'HARSHITA@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(9, 'ISHAAN', 'REDDY', 32,  
70000, 'ISHAAN@GCOMPANY.IN');
```

```
INSERT INTO EMPLOYEE VALUES(10, 'JANU', 'MUKHERJEE', 30,  
53000, 'JANU@GCOMPANY.IN');
```

## FSD Training Program

```
SELECT COUNT(*) FROM EMPLOYEE;
```

```
+-----+
```

```
| COUNT(*) |
```

```
+-----+
```

```
|      40 |
```

```
+-----+
```

```
1 row in set (0.01 sec)
```

```
SELECT * FROM EMPLOYEE;
```

```
SET @row_number = 0;
```

```
SELECT ID, FIRST_NAME, LAST_NAME, AGE, EMAIL, SALARY,  
(@row_number:=@row_number + 1) AS ROWNUM FROM EMPLOYEE;
```

SET @row\_number = 0; initializes a user-defined variable @row\_number and sets it to 0.

(@row\_number:=@row\_number + 1) AS ROWNUM increments the @row\_number variable for each row, effectively assigning a row number to each result.

```
SELECT ID, FIRST_NAME, LAST_NAME, AGE, EMAIL,  
SALARY, (@row_number:=@row_number + 1) AS ROWNUM FROM  
EMPLOYEE;
```

```
+-----+-----+-----+-----+-----+-----+-----+
```

```
| ID | FIRST_NAME | LAST_NAME | AGE | EMAIL | SALARY | ROWNUM |
```

```
+-----+-----+-----+-----+-----+-----+-----+
```

```
| 1 | ARUN | PATEL | 22 | ARUN@GCOMPANY.IN | 40000 | 1 |
```

```
| 2 | BHAVESH | SHARMA | 24 | BHAVESH@GCOMPANY.IN | 30000 | 2 |
```

```
| 3 | CHAITANYA | SINGH | 23 | CHAITANYA@GCOMPANY.IN | 50000 | 3 |
```

```
| 4 | DEEPIKA | GUPTA | 26 | DEEPIKA@GCOMPANY.IN | 55000 | 4 |
```

```
| 5 | DHANUSH | KUMAR | 25 | DHANUSH@GCOMPANY.IN | 20000 | 5 |
```

FSD Training Program

	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		6	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		7	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		8	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		9	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		10	
	1		ARUN		PATEL		22		ARUN@GCOMPANY.IN		40000		11	
	2		BHAVESH		SHARMA		24		BHAVESH@GCOMPANY.IN		30000		12	
	3		CHAITANYA		SINGH		23		CHAITANYA@GCOMPANY.IN		50000		13	
	4		DEEPIKA		GUPTA		26		DEEPIKA@GCOMPANY.IN		55000		14	
	5		DHANUSH		KUMAR		25		DHANUSH@GCOMPANY.IN		20000		15	
	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		16	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		17	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		18	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		19	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		20	
	1		ARUN		PATEL		22		ARUN@GCOMPANY.IN		40000		21	
	2		BHAVESH		SHARMA		24		BHAVESH@GCOMPANY.IN		30000		22	
	3		CHAITANYA		SINGH		23		CHAITANYA@GCOMPANY.IN		50000		23	
	4		DEEPIKA		GUPTA		26		DEEPIKA@GCOMPANY.IN		55000		24	
	5		DHANUSH		KUMAR		25		DHANUSH@GCOMPANY.IN		20000		25	
	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		26	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		27	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		28	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		29	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		30	
	1		ARUN		PATEL		22		ARUN@GCOMPANY.IN		40000		31	
	2		BHAVESH		SHARMA		24		BHAVESH@GCOMPANY.IN		30000		32	
	3		CHAITANYA		SINGH		23		CHAITANYA@GCOMPANY.IN		50000		33	
	4		DEEPIKA		GUPTA		26		DEEPIKA@GCOMPANY.IN		55000		34	
	5		DHANUSH		KUMAR		25		DHANUSH@GCOMPANY.IN		20000		35	
	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		36	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		37	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		38	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		39	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		40	

## FSD Training Program

+-----+-----+-----+-----+-----+-----+-----+-----+-----+

40 rows in set, 1 warning (0.00 sec)

### PAGINATION IN MYSQL

Even if we have so many records in a table what if I want to display a particular number of records. In such case we can use pagination concept. In case of Oracle We have rownum but in mysql we don't have that.

SET @row\_number = 0;

Query OK, 0 rows affected (0.00 sec)

SELECT ID, FIRST\_NAME, LAST\_NAME, AGE, EMAIL,  
SALARY, (@row\_number:=@row\_number + 1) AS RN FROM EMPLOYEE  
ORDER BY ID;

+-----+-----+-----+-----+-----+-----+-----+-----+							
ID	FIRST_NAME	LAST_NAME	AGE	EMAIL	SALARY	RN	
+-----+-----+-----+-----+-----+-----+-----+-----+							
1	ARUN	PATEL	22	ARUN@GCOMPANY.IN	40000	1	
1	ARUN	PATEL	22	ARUN@GCOMPANY.IN	40000	2	
1	ARUN	PATEL	22	ARUN@GCOMPANY.IN	40000	3	
1	ARUN	PATEL	22	ARUN@GCOMPANY.IN	40000	4	
2	BHAVESH	SHARMA	24	BHAVESH@GCOMPANY.IN	30000	5	
2	BHAVESH	SHARMA	24	BHAVESH@GCOMPANY.IN	30000	6	
2	BHAVESH	SHARMA	24	BHAVESH@GCOMPANY.IN	30000	7	
2	BHAVESH	SHARMA	24	BHAVESH@GCOMPANY.IN	30000	8	
3	CHAITANYA	SINGH	23	CHAITANYA@GCOMPANY.IN	50000	9	
3	CHAITANYA	SINGH	23	CHAITANYA@GCOMPANY.IN	50000	10	
3	CHAITANYA	SINGH	23	CHAITANYA@GCOMPANY.IN	50000	11	
3	CHAITANYA	SINGH	23	CHAITANYA@GCOMPANY.IN	50000	12	
4	DEEPIKA	GUPTA	26	DEEPIKA@GCOMPANY.IN	55000	13	
4	DEEPIKA	GUPTA	26	DEEPIKA@GCOMPANY.IN	55000	14	
4	DEEPIKA	GUPTA	26	DEEPIKA@GCOMPANY.IN	55000	15	
4	DEEPIKA	GUPTA	26	DEEPIKA@GCOMPANY.IN	55000	16	

## FSD Training Program

	5		DHANUSH		KUMAR		25		DHANUSH@GCOMPANY.IN		20000		17	
	5		DHANUSH		KUMAR		25		DHANUSH@GCOMPANY.IN		20000		18	
	5		DHANUSH		KUMAR		25		DHANUSH@GCOMPANY.IN		20000		19	
	5		DHANUSH		KUMAR		25		DHANUSH@GCOMPANY.IN		20000		20	
	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		21	
	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		22	
	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		23	
	6		EKTA		YADAV		28		YADAV@GCOMPANY.IN		35000		24	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		25	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		26	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		27	
	7		GAURAV		RAO		21		GAURAV@GCOMPANY.IN		60000		28	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		29	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		30	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		31	
	8		HARSHITA		REDDY		29		HARSHITA@GCOMPANY.IN		56000		32	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		33	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		34	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		35	
	9		ISHAAN		REDDY		32		ISHAAN@GCOMPANY.IN		70000		36	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		37	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		38	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		39	
	10		JANU		MUKHERJEE		30		JANU@GCOMPANY.IN		53000		40	

+-----+-----+-----+-----+-----+-----+-----+-----+

40 rows in set, 1 warning (0.00 sec)

### LIMIT:

LIMIT is used to restrict the number of rows returned by a query.

It takes one or two arguments: LIMIT x or LIMIT x, y.

x specifies the maximum number of rows to return.

y (optional) specifies the offset or the number of rows to skip before starting to return rows.

Retrieve the first 5 rows from a table.

## FSD Training Program

```
SELECT * FROM EMPLOYEE LIMIT 5;
```

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 1     | ARUN       | PATEL     | 22   | 40000  | ARUN@GCOMPANY.IN                  |
| 2     | BHAVESH    | SHARMA    | 24   | 30000  | BHAVESH@GCOMPANY.IN              |
| 3     | CHAITANYA  | SINGH     | 23   | 50000  | CHAITANYA@GCOMPANY.IN            |
| 4     | DEEPIKA    | GUPTA     | 26   | 55000  | DEEPIKA@GCOMPANY.IN              |
| 5     | DHANUSH    | KUMAR     | 25   | 20000  | DHANUSH@GCOMPANY.IN              |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

LIMIT 5, 5

First value: Specifies the offset, which is the number of rows to skip before starting to retrieve data.

Second value: Specifies the limit, which is the maximum number of rows to retrieve after the offset(first number).

Retrieve rows 6 through 10 from a table.

```
SELECT * FROM EMPLOYEE LIMIT 5, 5;
```

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 6     | EKTA       | YADAV     | 28   | 35000  | YADAV@GCOMPANY.IN                |
| 7     | GAURAV     | RAO       | 21   | 60000  | GAURAV@GCOMPANY.IN              |
| 8     | HARSHITA   | REDDY     | 29   | 56000  | HARSHITA@GCOMPANY.IN            |
| 9     | ISHAAN     | REDDY     | 32   | 70000  | ISHAAN@GCOMPANY.IN              |
| 10    | JANU       | MUKHERJEE | 30   | 53000  | JANU@GCOMPANY.IN                 |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

### OFFSET:

OFFSET is used to skip a specified number of rows before starting to return rows.

It's usually used in combination with LIMIT.



## FSD Training Program

The OFFSET value starts from 0.

Skip the first 3 rows and return the next 5.

```
SELECT * FROM EMPLOYEE LIMIT 5 OFFSET 3;
```

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 4     | DEEPIKA   | GUPTA     | 26   | 55000  | DEEPIKA@GCOMPANY.IN              |
| 5     | DHANUSH   | KUMAR     | 25   | 20000  | DHANUSH@GCOMPANY.IN              |
| 6     | EKTA      | YADAV     | 28   | 35000  | YADAV@GCOMPANY.IN                |
| 7     | GAURAV    | RAO       | 21   | 60000  | GAURAV@GCOMPANY.IN              |
| 8     | HARSHITA  | REDDY     | 29   | 56000  | HARSHITA@GCOMPANY.IN            |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

Alternatively, you can use the shorter form LIMIT x, y where x is the offset and y is the number of rows to return.

Skip the first 2 rows and return the next 8.

```
SELECT * FROM EMPLOYEE LIMIT 2, 8;
```

```
+-----+-----+-----+-----+-----+-----+
| ID    | FIRST_NAME | LAST_NAME | AGE  | SALARY | EMAIL                                |
+-----+-----+-----+-----+-----+-----+
| 3     | CHAITANYA | SINGH     | 23   | 50000  | CHAITANYA@GCOMPANY.IN            |
| 4     | DEEPIKA   | GUPTA     | 26   | 55000  | DEEPIKA@GCOMPANY.IN              |
| 5     | DHANUSH   | KUMAR     | 25   | 20000  | DHANUSH@GCOMPANY.IN              |
| 6     | EKTA      | YADAV     | 28   | 35000  | YADAV@GCOMPANY.IN                |
| 7     | GAURAV    | RAO       | 21   | 60000  | GAURAV@GCOMPANY.IN              |
| 8     | HARSHITA  | REDDY     | 29   | 56000  | HARSHITA@GCOMPANY.IN            |
| 9     | ISHAAN    | REDDY     | 32   | 70000  | ISHAAN@GCOMPANY.IN              |
| 10    | JANU      | MUKHERJEE | 30   | 53000  | JANU@GCOMPANY.IN                 |
+-----+-----+-----+-----+-----+-----+
8 rows in set (0.00 sec)
```

# FSD Training Program

## TO FETCH ALTERNATIVE RECORDS FROM A TABLE

```
SELECT * FROM EMPLOYEE WHERE MOD(id, 2) = 0;
```

ID	FIRST_NAME	LAST_NAME	AGE	SALARY	EMAIL
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN
2	BHAVESH	SHARMA	24	30000	BHAVESH@GCOMPANY.IN
4	DEEPIKA	GUPTA	26	55000	DEEPIKA@GCOMPANY.IN
6	EKTA	YADAV	28	35000	YADAV@GCOMPANY.IN
8	HARSHITA	REDDY	29	56000	HARSHITA@GCOMPANY.IN
10	JANU	MUKHERJEE	30	53000	JANU@GCOMPANY.IN

20 rows in set (0.00 sec)

MOD(id, 2) calculates the remainder when id is divided by 2. This will be 0 for even ids and 1 for odd ids.

WHERE MOD(id, 2) = 0 filters the rows to only include those where the id is even.