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 IDENTIERS 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	CREATE Creates a new table, a view of a table, or other object in the database.
2	ALTER Modifies an existing database object, such as a table.
3	DROP Deletes an entire table, a view of a table or other objects in the database.

2. DML - Data Manipulation Language

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

3.DCL - Data Control Language

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

4. <u>DQL - Data Query Language</u>

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

4	WHERE 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	ORDER BY 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	GROUP BY The GROUP BY clause is used to group rows with the same values in one or more columns into summary rows.
7	HAVING 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

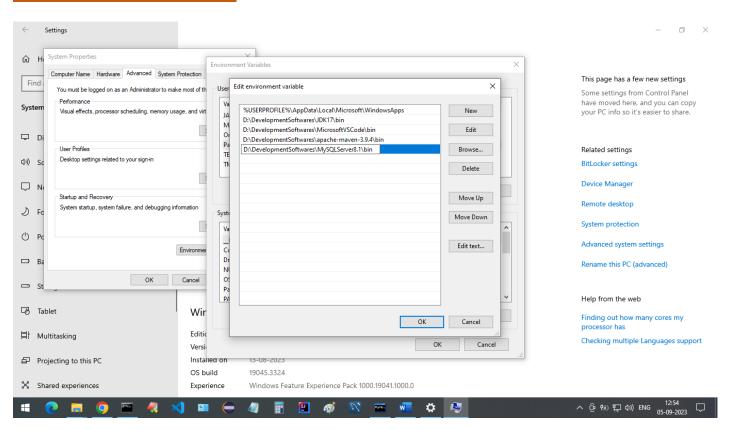
Field: StudentId, StudentName, StudentMarks

<u>Please follow the link to Learn how download and</u> 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 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?

```
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.

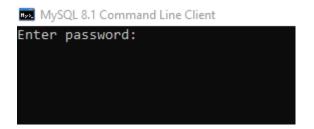
Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

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 chage the user with the following command.

SYSTEM mysql -u vijay -p;

Enter password: *****

Alternatively you can use

\! mysql -u vijay -p

Enter password: *****

FSD Training Program

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
 mysq1
| 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 |
 mysq1
| mysql_notes
| 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 PRIVILIAGES TO THE USER?

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

HOW TO CHECK CURRENT USER PRIVILIAGES?

SHOW GRANTS FOR 'root'@'localhost';

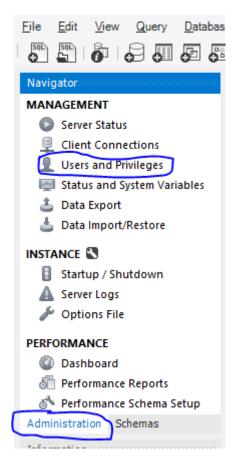
HOW TO FOR CHANGES TO TAKE EFFECT IMMEDIATELY?

FLUSH PRIVILEGES;

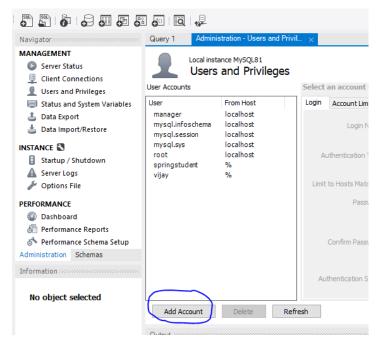
Note: Starting from MySQL 5.7.3, the FLUSH PRIVILEGES; statement is no longer strictly required after executing GRANT or REVOKE statements. The server automatically reloads the grant tables in these cases.

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



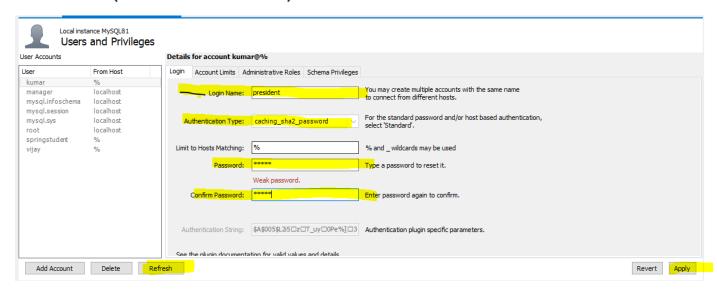
- 3. Click on Users and Privileges
- 4. Click on Add account to create a new account



5. Fill in the deatails

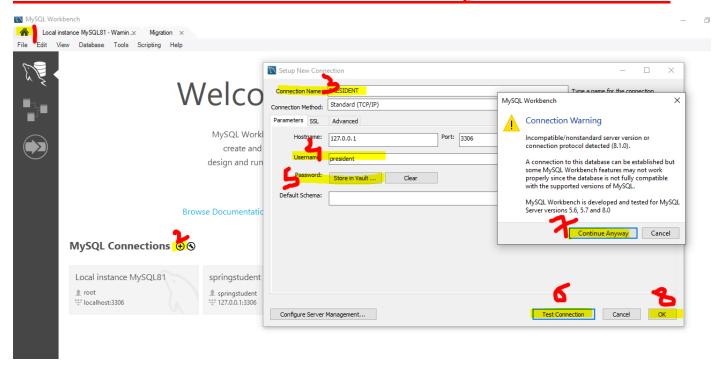
FSD Training Program

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

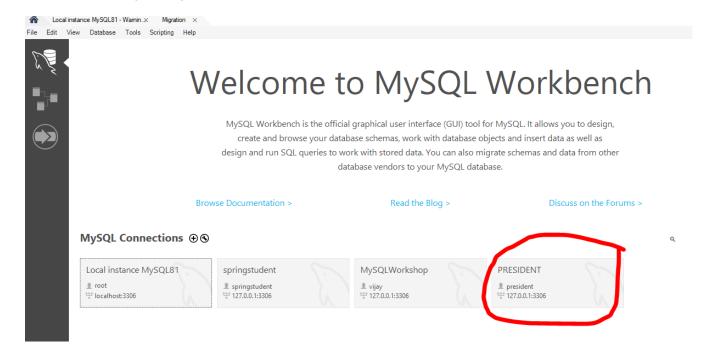


6.Click on Apply and Refresh

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



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
```

10 rows in set (0.00 sec)

```
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;
+----+
+----+
  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
+----+
```

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;
+----+
    | 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;
+----+
+----+
| 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
+----+
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)

Rows matched: 1 Changed: 1 Warnings: 0

```
SELECT * FROM STUDENT;
+----+
+----+
 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
```

FSD Training Program

```
SELECT * FROM STUDENT;
+----+
+----+
 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
+----+
```

WHAT IS 'NULL' IN SQL?

12 rows in set (0.00 sec)

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;

12 rows in set (0.00 sec)

+		.+.		.+.		.+.		•+
1	ID		FIRST_NAME	•	AGE	•		1
				· T ·				
ı	101		ARUN		20		CSE	ı
	102		BHAVESH		21		ISE	
1	103	I	CHAITANYA		22		ECE	
	104		DEEPIKA		23	I	MECH	I
	112	I	RAJAT		29		AUTOMOBILE	
1	106	I	EKTA	I	25		AI	I
1	107		GAURAV		26		ARCH	
I	108	I	RISHI	1	27	١	CHEMICAL	١
I	109	I	ISHAAN	1	28	١	CIVIL	١
	110	I	JANU		29	I	EEE	I
	110		JANU		29		EEE	
١	111		PRANAV		30	I	NULL	I
+		+		+-		+		+

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;
+----+
+----+
 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
```

12 rows in set (0.00 sec)

110 | JANU | 20 | EEE

111 | PRANAV | 20 | NULL

+----+

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 STATMENT?

```
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;
+----+
+----+
| 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 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;
+----+
+----+
| 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

```
FSD Training Program
 109 | ANANYA | 28 | CIVIL |
 110 | ANANYA | 29 | EEE
+----+
10 rows in set (0.00 sec)
HOW WOULD YOU UPDATE MULTILE 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;
+----+
+----+
| 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;
+----+
+----+
| 15 | ARUN | 42 | CSE |
| 15 | BHAVESH | 42 | ISE
| 15 | CHAITANYA | 42 | ECE
15 | ANANYA | 42 | DS
| 15 | ANANYA | 42 | ARCH |
| 15 | ANANYA | 42 | CHEMICAL |
 15 | ANANYA | 42 | CIVIL |
15 | ANANYA | 42 | EEE |
+----+
```

DELETE FROM STUDENT;
Query OK, 10 rows affected (0.01 sec)

10 rows in set (0.00 sec)

HOW WOULD YOU EXECUTE MULTIPLE STATMENTS 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.

```
PRESIDENT - Warning - not s... ×
Edit View Query Database Server Tools Scripting Help
RT INTO STUDENT VALUES (101, 'ARUN', 20, 'CSE');
  1 •
        INSERT INTO STUDENT VALUES (102, 'BHAVESH', 21, 'ISE');
  2 •
        INSERT INTO STUDENT VALUES (103, 'CHAITANYA', 22, 'ECE');
  3 •
  4 •
        INSERT INTO STUDENT VALUES (104, 'DEEPIKA', 23, 'MECH');
  5 •
        INSERT INTO STUDENT VALUES (105, 'DHANUSH', 24, 'DS');
  6 •
        INSERT INTO STUDENT VALUES (106, 'EKTA', 25, 'AI');
  7 •
        INSERT INTO STUDENT VALUES (107, 'GAURAV', 26, 'ARCH');
        INSERT INTO STUDENT VALUES (108, 'HARSHITA', 27, 'CHEMICAL');
  8 •
        INSERT INTO STUDENT VALUES (109, 'ISHAAN', 28, 'CIVIL');
  9 •
        INSERT INTO STUDENT VALUES (110, 'JANU', 29, 'EEE');
 10 •
 11
```

```
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;
+----+
+----+
| 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';
```

```
FSD Training Program
Query OK, 1 row affected (0.00 sec)
SELECT * FROM STUDENT;
+----+
+----+
| 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)

```
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');
```

CREATE TABLE EMPLOYEE (ID INTEGER, FIRST NAME VARCHAR(90),

SELECT * FROM EMPLOYEE;

+	+	_		-+-		-+		-+		+		t
1	[D	I	FIRST_NAME	I	LAST_NAME	I	AGE	I	SALARY	I	EMAIL	
+	+	_		-+-		-+		-+		+		۲
	1		ARUN		PATEL		22	I	40000		ARUN@GCOMPANY.IN	
1	2		BHAVESH		SHARMA	١	24	I	30000	I	BHAVESH@GCOMPANY.IN	l
1	3		CHAITANYA	I	SINGH	I	23	١	50000	١	CHAITANYA@GCOMPANY.IN	l
1	4		DEEPIKA	I	GUPTA	1	26	I	55000	I	DEEPIKA@GCOMPANY.IN	l
1	5		DHANUSH	I	KUMAR	١	25	I	20000	I	DHANUSH@GCOMPANY.IN	l
1	6		EKTA	I	YADAV	١	28	I	35000	I	YADAV@GCOMPANY.IN	l
1	7		GAURAV	I	RAO	١	21	I	60000	I	GAURAV@GCOMPANY.IN	l
1	8		HARSHITA	I	REDDY	١	29	I	56000	I	HARSHITA@GCOMPANY.IN	l
1	9		ISHAAN	I	REDDY	١	32	I	70000	I	ISHAAN@GCOMPANY.IN	l
ı	10		JANU	I	MUKHERJEE	I	30	I	53000	I	JANU@GCOMPANY.IN	l
+	+			-+-		-+		-+		.+		+

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?

SI	SELECT * FROM EMPLOYEE WHERE ID > 5;												
+-		+-		-+-		-+		+-		+-		-+	
-	ID	1	FIRST_NAME		LAST_NAME	A	GE	I	SALARY	I	EMAIL		
+-		+-		-+-		-+		+-		+-		-+	
I	6	I	EKTA	I	YADAV	I	28	I	35000	I	YADAV@GCOMPANY.IN	I	
Ι	7		GAURAV	Ι	RAO	1	21	I	60000	I	GAURAV@GCOMPANY.IN	I	
1	8	I	HARSHITA	I	REDDY	I	29	Ī	56000	I	HARSHITA@GCOMPANY.IN	Ī	
1	9	I	ISHAAN	I	REDDY	1	32	Ī	70000	I	ISHAAN@GCOMPANY.IN	Ī	
Ι	10		JANU	Ι	MUKHERJEE	1	30	I	53000	I	JANU@GCOMPANY.IN	I	
+-		+-		-+-		-+		+-		+-		-+	
5	rows	in	set (0.00	se	c)								

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?**

SE	LEC	T	* FROM	EMP	PLOYEE W	IHE	RE A	GE	NOT	BETWEEN	22 ANI	28;
+		+-		-+-		+-		+-		+		+
:	ID	I	FIRST_NAME	:	LAST_NAME		AGE	:	SALARY	EMAIL		
+		+-		-+-		+-		+-		+		+
I	7	I	GAURAV	1	RAO		21	I	60000	GAURAV@G	COMPANY.	IN
I	8	I	HARSHITA	1	REDDY		29	I	56000	HARSHITA	@GCOMPAN	Y.IN
I	9	I	ISHAAN	1	REDDY		32	I	70000	ISHAAN@G	COMPANY.	IN
I	10	1	JANU	1	MUKHERJEE	 	30		53000	JANU@GCO	MPANY.IN	1
+		+-		-+-		+-		+-		+		+
4 ı	rows	in	set (0.00	se	c)							

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	1	LAST_NAME	1	AGE	I	SALARY	I	EMAIL
+	+-		-+-		-+-		+-		+	+
I	1	ARUN	Ι	PATEL	1	22	I	40000	I	ARUN@GCOMPANY.IN
I	4	DEEPIKA	Ι	GUPTA	1	26	I	55000	I	DEEPIKA@GCOMPANY.IN
I	9	ISHAAN	Ι	REDDY	1	32	I	70000	I	ISHAAN@GCOMPANY.IN
+	+-		-+-		-+-		+-		٠+	+

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	1	FIRST_NAME	İ	LAST_NAME	/	AGE	I	SALARY	1		·+
		BHAVESH CHAITANYA DHANUSH EKTA GAURAV HARSHITA JANU	1	SHARMA SINGH KUMAR YADAV RAO REDDY MUKHERJEE		24 23 25 28 21 29 30		30000 50000 20000 35000		YADAV@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%';

+	+-		-+		+		+		+		٠+
ID	I	FIRST_NAME	I	LAST_NAME	I	AGE	I	SALARY	I	EMAIL	I
+	+-		-+		+		+		+		+
1	1	ARUN	Ī	PATEL	I	22	I	40000	I	ARUN@GCOMPANY.IN	I
1	7	GAURAV	1	RAO	١	21	I	60000	I	GAURAV@GCOMPANY.IN	I
1	8	HARSHITA	I	REDDY	١	29	I	56000	I	HARSHITA@GCOMPANY.IN	I
+	+-		-+		+		+		+		-+

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?

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

1 row in set (0.00 sec)

FSD Training Program

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	I								
	HARSHITA		29	I								
	ISHAAN		32	I								
	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?

SEL	LECT FIRST_	NAME	, A	GE	E, EMAIL FROM EMPLOYEE;))
+		+		+-		-+
F	IRST_NAME	A(ŝΕ		EMAIL	
+		+		+-		+
l	ARUN	1	22	I	ARUN@GCOMPANY.IN	I
	BHAVESH		24		BHAVESH@GCOMPANY.IN	1
l	CHAITANYA	1	23	I	CHAITANYA@GCOMPANY.IN	I
	DEEPIKA		26	I	DEEPIKA@GCOMPANY.IN	
1	DHANUSH		25		DHANUSH@GCOMPANY.IN	I
l	EKTA	1	28	I	YADAV@GCOMPANY.IN	1
l	GAURAV	1	21	I	GAURAV@GCOMPANY.IN	1
	HARSHITA		29	I	HARSHITA@GCOMPANY.IN	
	ISHAAN		32	I	ISHAAN@GCOMPANY.IN	
l	JANU	1	30	I	JANU@GCOMPANY.IN	I
+		+		+-		-+
10	rows in se	et (6	9.00	9	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	1
+		+		+		-+
	ARUN		22		PATEL	
I	BHAVESH		24		SHARMA	I
	CHAITANYA	I	23	I	SINGH	
I	DEEPIKA	I	26	I	GUPTA	
1	DHANUSH	I	25	1	KUMAR	
	EKTA		28		YADAV	
	GAURAV		21		RAO	
1	HARSHITA		29		REDDY	
1	ISHAAN		32		REDDY	
	JANU	I	30		MUKHERJEE	
+		+		+		-+
	• • • • • • • • • • • • • • • • • • • •		(0.00)			

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;

+		+	+-		-+
I	NAME	EMPLOYEE_AGE	I	LAST_NAME	
+		+	+-		-+
	ARUN	22	I	PATEL	1
1	BHAVESH	24	I	SHARMA	
I	CHAITANYA	23	I	SINGH	1
1	DEEPIKA	26		GUPTA	
1	DHANUSH	25		KUMAR	
I	EKTA	28	I	YADAV	1
1	GAURAV	21		RAO	
1	HARSHITA	29		REDDY	1
1	ISHAAN	32		REDDY	
1	JANU	30	I	MUKHERJEE	
+		+	-+-		-+
10	o rows in se	t (0.00 sec)			

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.

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 |

+------+

1 row in set (0.00 sec)
```

```
EXAMPLE: 3
SELECT COUNT(*) "RECORDS COUNT" FROM EMPLOYEE;
+-----
| RECORDS COUNT |
+----+
           10 l
+----+
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 l
+----+
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 ;
--------	-------	-------	-------	------	-------------------

+		·+
I	MAX(FIRST_NAME)	1
+		+
I	JANU	1
+		+
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;

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

FSD Training Program	
7 GAURAV RAO 21 60000 GAURAV@GCC	OMPANY.IN
8 HARSHITA REDDY 29 56000 HARSHITA@G	GCOMPANY.IN
9 ISHAAN	OMPANY.IN
10 JANU	•
++	•
10 rows in set (0.00 sec)	•
·	THE
ORDER EMPLOYEE RECORDS USING ORDER BY IN	
ASCENDING ORDER BY CONSIDERING FIRST NAME	E COLUMN
USING AS KEYWORD?	
SELECT * FROM EMPLOYEE ORDER BY FIRST_NAME AS	C;
+	+
ID	1
+	+
1 ARUN PATEL 22 40000 ARUN@GCOMPANY.	IN
2 BHAVESH SHARMA 24 30000 BHAVESH@GCOMPAN	Y.IN
3 CHAITANYA SINGH 23 50000 CHAITANYA@GCOMP	ANY.IN
4 DEEPIKA GUPTA 26 55000 DEEPIKA@GCOMPAN	Y.IN
5 DHANUSH KUMAR 25 20000 DHANUSH@GCOMPAN	Y.IN
6 EKTA YADAV 28 35000 YADAV@GCOMPANY.	IN
7 GAURAV RAO 21 60000 GAURAV@GCOMPANY	.IN
8 HARSHITA REDDY 29 56000 HARSHITA@GCOMPA	NY.IN
9 ISHAAN REDDY 32 70000 ISHAAN@GCOMPANY	.IN
10 JANU MUKHERJEE 30 53000 JANU@GCOMPANY.I	N
10 rows in set (0.00 sec)	+
ORDER EMPLOYEE RECORDS USING ORDER BY IN	THE
DESCENDING ORDER BY CONSIDERING FIRST NAME OF THE PROPERTY OF	
SELECT * FROM EMPLOYEE ORDER BY FIRST_NAME DES	SC;
+	+
ID	1
++	+

	FSD	Training	Program
--	------------	----------	---------

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

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;

+		+-		-+-		-+		-+			+	+
1	D	I	FIRST_NAME	I	LAST_NAME	I	AGE	I		SALARY	I	EMAIL
+		+-		-+-		-+		-+	-		+	+
1	7	I	GAURAV	I	RAO	-	21	I		60000	I	GAURAV@GCOMPANY.IN
1	1	I	ARUN	I	PATEL	-	22	١		40000	I	ARUN@GCOMPANY.IN
1	3	I	CHAITANYA	I	SINGH	-	23	١		50000	I	CHAITANYA@GCOMPANY.IN
1	2	I	BHAVESH	I	SHARMA	-	24	١		30000	I	BHAVESH@GCOMPANY.IN
1	5	I	DHANUSH	I	KUMAR	-	25	١		20000	I	DHANUSH@GCOMPANY.IN
I	4	I	DEEPIKA		GUPTA	I	26	١		55000	I	DEEPIKA@GCOMPANY.IN
I	6	I	EKTA		YADAV	I	28	١		35000	I	YADAV@GCOMPANY.IN
I	8	I	HARSHITA		REDDY	I	29	١		56000	I	HARSHITA@GCOMPANY.IN
I	10	I	JANU		MUKHERJEE	I	30	١		53000	I	JANU@GCOMPANY.IN
I	9	I	ISHAAN	I	REDDY	I	32	١		70000	I	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;

+		-+		-+		-+		+		+		+
١	ID	I	FIRST_NAME	I	LAST_NAME	I	AGE	I	SALARY	I	EMAIL	I
+		-+		-+		+		+		+		+
I	5	I	DHANUSH	١	KUMAR	I	25	I	20000	Ī	DHANUSH@GCOMPANY.IN	I
١	2	١	BHAVESH	١	SHARMA	I	24	I	30000	I	BHAVESH@GCOMPANY.IN	I
I	6	١	EKTA	١	YADAV	١	28	I	35000	١	YADAV@GCOMPANY.IN	I
1	1	١	ARUN		PATEL	١	22	I	40000	١	ARUN@GCOMPANY.IN	
1	3	١	CHAITANYA		SINGH	١	23	I	50000	١	CHAITANYA@GCOMPANY.IN	I
1	10	١	JANU		MUKHERJEE	١	30	I	53000	١	JANU@GCOMPANY.IN	I
I	4	I	DEEPIKA		GUPTA	١	26	I	55000	I	DEEPIKA@GCOMPANY.IN	I
1	8	١	HARSHITA		REDDY	١	29	I	56000	١	HARSHITA@GCOMPANY.IN	I
١	7	I	GAURAV	١	RAO	I	21	I	60000	Ī	GAURAV@GCOMPANY.IN	I
I	9	I	ISHAAN	١	REDDY	I	32	I	70000	I	ISHAAN@GCOMPANY.IN	Ī
+		-+		-+		-+		+		+		+
			_									

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.

FS	D Trai	ning Progr	am	1					
ı	1	ARUN	ı	PATEL		22	40000 ARUN@GCOMPANY	.IN	I
I	3	CHAITANYA	ı	SINGH	I	23	50000 CHAITANYA@GCOMF	PANY.IN	I
I	2	BHAVESH	ı	SHARMA	I	24	30000 BHAVESH@GCOMPAN	NY.IN	I
1	5	DHANUSH	I	KUMAR	I	25	20000 DHANUSH@GCOMPAN	NY.IN	I
1	4	DEEPIKA	I	GUPTA	I	26	55000 DEEPIKA@GCOMPAN	NY.IN	I
1	6	EKTA	I	YADAV	I	28	35000 YADAV@GCOMPANY	.IN	1
I	8	HARSHITA	I	REDDY	I	29	56000 HARSHITA@GCOMPA	ANY.IN	1
I	10	JANU	I	MUKHERJEE	I	30	53000 JANU@GCOMPANY.	EN	I
I	9	ISHAAN	ı	REDDY	I	32	70000 ISHAAN@GCOMPANY	Y.IN	I
I	11	ARUL	ı	PATEL	I	35	40000 ARUL@GCOMPANY	.IN	I
I	12	ADITI	I	PATEL	I	35	60000 ADITI@GCOMPANY	Y.IN	1
+	+-		-+-		-+	+-			+
12	rows i	n set (0.00	ð s	ec)					
TN	CEDT	TNTO EME) (NEE VALI	IEC	(11	'ADIII' 'DATEI'	25 <i>10</i>	1000
					JES	(11,	'ARUL', 'PATEL',	35, 40	9000,
'Α	RUL@G	COMPANY.	I	۱');				-	0000,
'A IN	RUL@G SERT	COMPANY.	IN PLO	N'); DYEE VALU	JES	(12,	'ARUL', 'PATEL',	-	9000,
'A IN	RUL@G SERT	COMPANY.	IN PLO	N'); DYEE VALU	JES	(12,		-	9000,
'A IN	RUL@G SERT	COMPANY.	IN PLO	N'); DYEE VALU	JES	(12,		-	9000,
'A IN 60	RUL@G SERT 000,	COMPANY. INTO EMF 'ADITI@C	IN PLO GCO	N'); DYEE VALU DMPANY.IM	JES N')	(12, ;		35,	9000,
'A' IN: 600	RUL@G SERT 000, <u>DER</u>	INTO EMF 'ADITI@G	IN PLO GCO	N'); DYEE VALU DMPANY.IN	JES N')	(12, ; <u>SING</u>	'ADITI', 'PATEL',	35,	9000,
'A' IN: 600	RUL@G SERT 000, DER CEND	INTO EMF 'ADITI@C	E ER	N'); DYEE VALU DMPANY.IN RECORDS	JES N')	(12, ; <u>SING</u> RY I	'ADITI', 'PATEL', ORDER BY AND AG	35, GE IN	9000,
'A' IN: 600	RUL@G SERT 000, DER CEND	INTO EMF 'ADITI@G EMPLOYE ING ORD * FROM E	E ER	N'); DYEE VALU DMPANY.IN RECORDS R AND SA PLOYEE OF	JES N') S U NLA RDE	(12,; SING RY I R BY	'ADITI', 'PATEL', ORDER BY AND AG N DESC ORDER?	35, GE IN ESC;	
'A' IN: 600 OR AS SE No:	RUL@G SERT 000, DER CEND LECT	INTO EMF 'ADITI@G EMPLOYE ING ORD * FROM E	E ER	N'); DYEE VALU DMPANY.IN RECORDS R AND SA PLOYEE OF	JES N') S U ALA RDE	(12,; SING RY I R BY irst	'ADITI', 'PATEL', ORDER BY AND AG N DESC ORDER? AGE ASC, SALARY D	35, SE IN ESC; in the	
OR AS SE	RUL@G SERT 000, DER CEND LECT te: Fecifi	INTO EMF 'ADITI@G EMPLOYE ING ORD * FROM E irst corder	E ER	N'); DYEE VALU DMPANY.IN RECORDS R AND SA PLOYEE OF iders the if two v	JES N') GU LA RDE e f	(12,; SING RY I R BY irst ues c	'ADITI', 'PATEL', ORDER BY AND AG N DESC ORDER? AGE ASC, SALARY D column and sorts	35, ESC; in the n are	e the
OR AS SE	RUL@G SERT 000, DER CEND LECT te: Fecifi me th	INTO EMF 'ADITI@G EMPLOYE ING ORD * FROM E irst corder	E ER	N'); DYEE VALU DMPANY.IN RECORDS R AND SA PLOYEE OF iders the if two v	JES N') GU LA RDE e f	(12,; SING RY I R BY irst ues c	'ADITI', 'PATEL', ORDER BY AND AG DESC ORDER? AGE ASC, SALARY D column and sorts f the first column	35, ESC; in the n are	e the
OR AS SE No spi	RUL@G SERT 000, DER CEND LECT te: Fecifi me the	INTO EMF 'ADITI@C EMPLOYE ING ORD * FROM E irst corder led order led order	E ER	N'); DYEE VALU DMPANY.IN RECORDS R AND SA PLOYEE OF iders the if two visiders the	JES N') S U ALA RDE e f val	(12, ; SING RY I R BY irst ues c	'ADITI', 'PATEL', ORDER BY AND AG DESC ORDER? AGE ASC, SALARY D column and sorts f the first column	35, ESC; in the n are s in t	the the
OR AS SE No spessor spessor spessor spessor spessor spessor spessor specific specifi	RUL@G SERT 000, DER CEND LECT te: Fecifi me thecifi	INTO EMF 'ADITI@G EMPLOYE ING ORD * FROM E irst cor led order nen it co	E ER	N'); DYEE VALU DMPANY.IN RECORDS R AND SA PLOYEE OF iders the if two versiders the siders the	JES N') S U ALA RDE e f val	(12,; SING RY I R BY irst ues c secor	'ADITI', 'PATEL', ORDER BY AND AG DESC ORDER? AGE ASC, SALARY D column and sorts f the first column d column and sort	35, ESC; in the n are s in t	the the
'AIN' 600 OR AS SE No spi sai spi +	RUL@G SERT 000, DER CEND LECT te: Fecifi me th ecifi	INTO EMF 'ADITI@G EMPLOYE ING ORD * FROM E irst cor led order led order led order first_NAME	E ER EMF	N'); DYEE VALU DMPANY.IN RECORDS R AND SA PLOYEE OF iders the if two values siders the LAST_NAME	JES N') GUALA RDE e f val	(12,; SING RY I R BY irst ues c secor	'ADITI', 'PATEL', ORDER BY AND AG DESC ORDER? AGE ASC, SALARY D column and sorts f the first column d column and sort	35, ESC; in the n are s in t	the the

22 |

40000 | ARUN@GCOMPANY.IN

23 | 50000 | CHAITANYA@GCOMPANY.IN |

| PATEL

1 | ARUN

3 | CHAITANYA | SINGH

```
BHAVESH
   2 |
                           24 | 30000 | BHAVESH@GCOMPANY.IN
                SHARMA
   5 DHANUSH
                        25 | 20000 | DHANUSH@GCOMPANY.IN
              KUMAR
   4 | DEEPIKA | GUPTA
                        26 | 55000 | DEEPIKA@GCOMPANY.IN
   6 | EKTA | YADAV
                       | 28 | 35000 | YADAV@GCOMPANY.IN
                          29 |
                              56000 | HARSHITA@GCOMPANY.IN
   8 | HARSHITA | REDDY
  10 | JANU
             | MUKHERJEE |
                          30 |
                              53000 | JANU@GCOMPANY.IN
  9 | ISHAAN
                          32 | 70000 | ISHAAN@GCOMPANY.IN
             REDDY
  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 colum minimum value
7 | GAURAV | RAO | 21 | 60000 | GAURAV@GCOMPANY.IN
                      | 22 | 40000 | ARUN@GCOMPANY.IN
   1 ARUN
            PATEL
   3 | CHAITANYA | SINGH
                          23 |
                              50000 | CHAITANYA@GCOMPANY.IN |
   2 | BHAVESH
                          24 | 30000 | BHAVESH@GCOMPANY.IN
             SHARMA
   5 DHANUSH
              l kumar
                          25 l
                              20000 | DHANUSH@GCOMPANY.IN
             | GUPTA
   4 | DEEPIKA
                          26 l
                               55000 | DEEPIKA@GCOMPANY.IN
              | YADAV
   6 | EKTA
                          28
                               35000 | YADAV@GCOMPANY.IN
   8 | HARSHITA | REDDY
                          29 |
                               56000 | HARSHITA@GCOMPANY.IN
  10 | JANU
              | MUKHERJEE |
                               53000 | JANU@GCOMPANY.IN
                          30
   9 | ISHAAN
              REDDY
                          32
                               70000 | ISHAAN@GCOMPANY.IN
   13 | ARTI
              PATEL
                               10000 | ARTI@GCOMPANY.IN
                          35
   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(SALAY) 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
+----+
I 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	I	FIR	RST_NAME	L	AST_NAME	I	AGE	I	SALARY		·
I	2	ВН	IAVESH	:	SHARMA	I	24	I	30000		BHAVESH@GCOMPANY.IN
			set (0			.+-		-+-		+-	+

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;

+		-+		-+-		-+		-+		+		-+		+
: -		Ċ	FIRST_NAME		_	Ċ		Ċ		•		Ċ	RANK() OVER (ORDER BY SALARY DESC	:)
+-		-+		-+-		-+		-+		+		-+		
-	9	I	ISHAAN	Ι	REDDY	I	32	١	70000	I	ISHAAN@GCOMPANY.IN	١		1
I	7	I	GAURAV	I	RAO	ı	21	ı	60000	I	GAURAV@GCOMPANY.IN	I		2
ı	12	ı	ADITI	Ī	PATEL	ı	35	ı	60000	I	ADITI@GCOMPANY.IN	ı		2
ı	8	ı	HARSHITA	Ī	REDDY	I	29	ı	56000	ı	HARSHITA@GCOMPANY.IN	ı		4
I	4	I	DEEPIKA	ı	GUPTA	I	26	ı	55000	I	DEEPIKA@GCOMPANY.IN	ı		5
I	10	ı	JANU	I	MUKHERJEE	I	30	ı	53000	ı	JANU@GCOMPANY.IN	ı		6

FSD '	Traini	ng Progr	am	1						
3	CHAITA	NYA SINGH		23 50000	(CHAITANYA@G	GCOMPANY.IN			7
1	ARUN	PATEL		22 40000	I	ARUN@GCOMI	PANY.IN	I		8
11	ARUL	PATEL		35 40000	I	ARUL@GCOMI	PANY.IN	I		8
6	EKTA	YADAV		28 35000	'	YADAV@GCOMI	PANY.IN	I		10
2	BHAVES	H SHARMA	١			BHAVESH@GC		I		11
'	DHANUS	•		25 20000	-					12
		0.01 sec)		*						
RETF	RIEV	E THE	DE	TAILS ()F	THE	EMPL	ο،	YEE WITH THE	
									ID, FIRST NAME,	<u>) </u>
LAST	Γ NA	ME, AG	ìΕ,	SALARY	7 ,	AND	EMAI	L	?	
ELECT	* FROM	(SELECT I	D, I	FIRST NAME, L	.AS	T NAME,	AGE, SALA	ARY	_ , EMAIL, RANK() OVER(ORDER	R BY
									YEES WHERE RANKED_EMPLOYEE	
	-+				+-	+				+
ID	FIRS	T_NAME L	AST_	_NAME AGE	I	SALARY	EMAIL		RANKED_EMPLOYEE	S
	-+			+	+-	+				+
7	GAU	RAV I	RAO	21	I	60000	GAURAV@0	GCO	MPANY.IN	2
12	ADIT	I I	PATI	EL 35	I	60000	ADITI@G	GCO	MPANY.IN	2
	-+			+	+-	+				+
2 rows	in set	(0.00 sec)							
SELEC	T * ED	OM EMPLO	VEE							
SELEC	1 " FK	ON EMPLO	YEE	j						
 	+		-+-		-+	+		-+		+
ID	FI	RST_NAME	ı	LAST_NAME	ı	AGE	SALARY		EMAIL	ı
+·	+		-+-		-+	+		-+		+
:	1 A	RUN		PATEL		22	40000		ARUN@GCOMPANY.IN	-
:	2 B	HAVESH		SHARMA	1	24	30000		BHAVESH@GCOMPANY.IN	- [
:	3 C	HAITANYA	ı	SINGH	ı	23	50000		CHAITANYA@GCOMPANY.II	N
4	4 D	EEPIKA	l	GUPTA	Ī	26	55000		DEEPIKA@GCOMPANY.IN	Ī
	•)HANUSH	ı	KUMAR	i	25		·	•	'
	- I D	7.100		INVIIIALL	- 1					- 1
	- I -	'I/T A			i	20 l		·	-	
	·	KTA GAURAV	İ	YADAV RAO	İ	28 21	35000		YADAV@GCOMPANY.IN GAURAV@GCOMPANY.IN	

29 |

32 |

HARSHITA

ISHAAN

9 |

REDDY

REDDY

56000 | HARSHITA@GCOMPANY.IN

70000 | ISHAAN@GCOMPANY.IN

```
FSD Training Program
  10 | JANU | MUKHERJEE | 30 | 53000 | JANU@GCOMPANY.IN
                                                    11 | ARUL | PATEL | 35 | 40000 | ARUL@GCOMPANY.IN
1
  12 | ADITI | PATEL | 35 | 60000 | ADITI@GCOMPANY.IN
+----+
12 rows in set (0.00 sec)
DELETE FROM EMPLOYEE;
Ouery OK, 12 rows affected (0.01 sec)
INSERT THE SAME SET OF RECORDS 4 TIMES
INSERT INTO EMPLOYEE VALUES
  (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');
INSERT INTO EMPLOYEE VALUES
  (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');
INSERT INTO EMPLOYEE VALUES
  (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'),
```

```
FSD Training Program
  (10, 'JANU', 'MUKHERJEE', 30, 53000, 'JANU@GCOMPANY.IN');
INSERT INTO EMPLOYEE VALUES
  (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');
TOTAL NUMBER OF RECORDS
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;

т		г		Τ.		•		-т		- т		•		• т	
I	D	1	FIRST_NAME	I	LAST_NAME	I	AGE	I	EMAIL	I	SALARY	I	ROWNUM	I	
+		+		+-		+		-+		+		+		-+	
1	1	l	ARUN	I	PATEL	I	22	I	ARUN@GCOMPANY.IN	I	40000	I	1	I	
I	2	l	BHAVESH	I	SHARMA	I	24	I	BHAVESH@GCOMPANY.IN	I	30000	I	2	I	
I	3	l	CHAITANYA	I	SINGH		23	I	CHAITANYA@GCOMPANY.IN	I	50000	I	3	I	
I	4	l	DEEPIKA	I	GUPTA		26	I	DEEPIKA@GCOMPANY.IN	I	55000	I	4	I	
I	5	l	DHANUSH	I	KUMAR		25	I	DHANUSH@GCOMPANY.IN	I	20000	I	5	I	
I	6	l	EKTA	I	YADAV		28	I	YADAV@GCOMPANY.IN	I	35000	I	6	I	
1	7	l	GAURAV	I	RAO	I	21	I	GAURAV@GCOMPANY.IN	I	60000	I	7	I	
I	8	l	HARSHITA	I	REDDY	I	29	I	HARSHITA@GCOMPANY.IN	I	56000	I	8	١	
I	9	l	ISHAAN	I	REDDY	I	32	I	ISHAAN@GCOMPANY.IN	I	70000	I	9	١	
I	10	l	JANU	I	MUKHERJEE	I	30	I	JANU@GCOMPANY.IN	I	53000	I	10	I	
I	1	l	ARUN	I	PATEL	I	22	I	ARUN@GCOMPANY.IN	I	40000	I	11	١	
I	2	l	BHAVESH	I	SHARMA	I	24	I	BHAVESH@GCOMPANY.IN	I	30000	I	12	١	
I	3	l	CHAITANYA	I	SINGH	I	23	I	CHAITANYA@GCOMPANY.IN	I	50000	I	13	I	
1	4	l	DEEPIKA	ı	GUPTA	ı	26	ı	DEEPIKA@GCOMPANY.IN	ı	55000	ı	14	ı	

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

+----+

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	I	LAST_NAME	I	AGE	I	EMAIL	I	SALARY	RN	I
ŀ	1		ARUN	ŀ	PATEL	ŀ	22	İ	ARUN@GCOMPANY.IN	İ	40000		L
I	1	١	ARUN	I	PATEL	I	22	I	ARUN@GCOMPANY.IN	I	40000	2	2
I	1	١	ARUN	I	PATEL	I	22	I	ARUN@GCOMPANY.IN	I	40000	3	3
I	1	١	ARUN	I	PATEL	I	22	I	ARUN@GCOMPANY.IN	I	40000	4	1
I	2	١	BHAVESH	I	SHARMA	I	24	I	BHAVESH@GCOMPANY.IN	I	30000	5	5
I	2	١	BHAVESH	I	SHARMA	I	24	I	BHAVESH@GCOMPANY.IN	I	30000	6	5
I	2	١	BHAVESH	I	SHARMA	I	24	I	BHAVESH@GCOMPANY.IN	I	30000	7	7
I	2	١	BHAVESH	I	SHARMA	I	24	I	BHAVESH@GCOMPANY.IN	I	30000	8	3
I	3	١	CHAITANYA	I	SINGH	I	23	I	CHAITANYA@GCOMPANY.IN	I	50000	9)
I	3	١	CHAITANYA	I	SINGH	I	23	I	CHAITANYA@GCOMPANY.IN	I	50000	10)
I	3	١	CHAITANYA	I	SINGH	I	23	I	CHAITANYA@GCOMPANY.IN	I	50000	11	L
I	3	١	CHAITANYA	I	SINGH	I	23	I	CHAITANYA@GCOMPANY.IN	I	50000	12	2
I	4	١	DEEPIKA	I	GUPTA	I	26	I	DEEPIKA@GCOMPANY.IN	I	55000	13	3
	4	١	DEEPIKA	١	GUPTA	I	26	I	DEEPIKA@GCOMPANY.IN	I	55000	14	1
	4	١	DEEPIKA	١	GUPTA	I	26	I	DEEPIKA@GCOMPANY.IN	I	55000	15	5
	4	١	DEEPIKA	١	GUPTA	I	26	I	DEEPIKA@GCOMPANY.IN	I	55000	16	5
I	5	I	DHANUSH	I	KUMAR	I	25	١	DHANUSH@GCOMPANY.IN	I	20000	17	7
I	5	١	DHANUSH	I	KUMAR	I	25	I	DHANUSH@GCOMPANY.IN	I	20000	18	3
I	5	١	DHANUSH	I	KUMAR	I	25	I	DHANUSH@GCOMPANY.IN	I	20000	19)
ı	5	١	DHANUSH	١	KUMAR	I	25	١	DHANUSH@GCOMPANY.IN	I	20000	20)

FS	D Training P	rogram					
I	6 EKTA	YADAV	I	28 YADAV@GCOMPANY.IN	I	35000	21
I	6 EKTA	YADAV	1	28 YADAV@GCOMPANY.IN	1	35000	22
I	6 EKTA	YADAV	1	28 YADAV@GCOMPANY.IN	1	35000	23
I	6 EKTA	YADAV	Ι	28 YADAV@GCOMPANY.IN		35000	24
I	7 GAURAV	RAO	Ι	21 GAURAV@GCOMPANY.IN		60000	25
Ι	7 GAURAV	RAO	Ι	21 GAURAV@GCOMPANY.IN	I	60000	26
Ι	7 GAURAV	RAO	Ι	21 GAURAV@GCOMPANY.IN	I	60000	27
Ι	7 GAURAV	RAO	Ι	21 GAURAV@GCOMPANY.IN	I	60000	28
I	8 HARSHITA	REDDY	1	29 HARSHITA@GCOMPANY.IN	I	56000	29
I	8 HARSHITA	REDDY	I	29 HARSHITA@GCOMPANY.IN	I	56000	30
I	8 HARSHITA	REDDY	I	29 HARSHITA@GCOMPANY.IN	I	56000	31
I	8 HARSHITA	REDDY	I	29 HARSHITA@GCOMPANY.IN	I	56000	32
I	9 ISHAAN	REDDY	I	32 ISHAAN@GCOMPANY.IN	I	70000	33
Ι	9 ISHAAN	REDDY	Ι	32 ISHAAN@GCOMPANY.IN	I	70000	34
I	9 ISHAAN	REDDY	I	32 ISHAAN@GCOMPANY.IN	I	70000	35
I	9 ISHAAN	REDDY	I	32 ISHAAN@GCOMPANY.IN	I	70000	36
I	10 JANU	MUKHERJEE	-	30 JANU@GCOMPANY.IN	I	53000	37
Ι	10 JANU	MUKHERJEE		30 JANU@GCOMPANY.IN	I	53000	38
Ι	10 JANU	MUKHERJEE	:	30 JANU@GCOMPANY.IN	I	53000	39
I	10 JANU	MUKHERJEE	:	30 JANU@GCOMPANY.IN	I	53000	40

40 rows in set, 1 warning (0.00 sec)

MYSQL LIMIT KEYWORD

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

	SELECT * FROM EMPLOYEE LIMIT 5;														
+	++++++														
ID	ID														
+	+-		-+-		+-		+		+		-+				
	1	ARUN	1	PATEL	Ī	22	١	40000	١	ARUN@GCOMPANY.IN	I				
1	2	BHAVESH	1	SHARMA	I	24	I	30000	١	BHAVESH@GCOMPANY.IN	I				
	3	CHAITANYA	1	SINGH	I	23	I	50000	١	CHAITANYA@GCOMPANY.IN	I				
	4	DEEPIKA	1	GUPTA	I	26	I	55000	١	DEEPIKA@GCOMPANY.IN	I				
1	5	DHANUSH	I	KUMAR	I	25	I	20000	١	DHANUSH@GCOMPANY.IN	I				
+	+-		-+-		+-		+		+		-+				
5 ro	ws in	set (0.00	se	ec)											

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;

+	+-		-+-		-+-		+-		+		-+
1	D	FIRST_NAME	I	LAST_NAME	I	AGE	I	SALARY	I	EMAIL	I
+	+-		-+-		-+-		+-		+		-+
1	6	EKTA		YADAV	I	28	I	35000	١	YADAV@GCOMPANY.IN	
1	7	GAURAV	Ι	RAO	1	21	I	60000	١	GAURAV@GCOMPANY.IN	1
1	8	HARSHITA	I	REDDY	I	29		56000		HARSHITA@GCOMPANY.IN	
	9	ISHAAN		REDDY	I	32		70000		ISHAAN@GCOMPANY.IN	
	10	JANU		MUKHERJEE	I	30	I	53000	١	JANU@GCOMPANY.IN	
+	+-		-+-		-+-		+-		+		-+

5 rows in set (0.00 sec)

USING LIMIT WITH OFFSET

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

It's usually used in combination with LIMIT.

The OFFSET value starts from 0.

SKIP THE FIRST 3 ROWS AND RETURN THE NEXT 5

SELECT * FROM EMPLOYEE LIMIT 5 OFFSET 3;

ID	l	FIRST_NAME		LAST_NAME	I			SALARY			
+	 	DEEPIKA DHANUSH EKTA GAURAV	-+· 	GUPTA KUMAR YADAV RAO		26 25 28 21		55000 20000 35000 60000		DEEPIKA@GCOMPANY.IN DHANUSH@GCOMPANY.IN YADAV@GCOMPANY.IN GAURAV@GCOMPANY.IN 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			•		•			EMAIL	
Т			т-		-		т.		•		•		Т
		3		CHAITANYA		SINGH	1	23	1	50000	I	CHAITANYA@GCOMPANY.IN	I
I		4	I	DEEPIKA	I	GUPTA	Ī	26	Ī	55000	I	DEEPIKA@GCOMPANY.IN	I
ı		5	ı	DHANUSH	I	KUMAR	I	25	١	20000	I	DHANUSH@GCOMPANY.IN	I
ı		6	ı	EKTA	I	YADAV	I	28	I	35000	I	YADAV@GCOMPANY.IN	I
ı		7	ı	GAURAV	ı	RAO	ı	21	I	60000	I	GAURAV@GCOMPANY.IN	ı
ı		8	ı	HARSHITA	I	REDDY	I	29	I	56000	I	HARSHITA@GCOMPANY.IN	I
ı		9	ı	ISHAAN	I	REDDY	I	32	I	70000	I	ISHAAN@GCOMPANY.IN	I
ı	1	LØ	l	JANU	I	MUKHERJEE	ı	30	I	53000	I	JANU@GCOMPANY.IN	
+			+-		. + -		-+		.+		+		+

8 rows in set (0.00 sec)

HOW TO FETCH ALTERNATIVE RECORDS FROM A TABLE

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

+		+-		-+-		+		-+		+		-+
I	ID	l	FIRST_NAME	I	LAST_NAME	I	AGE	I	SALARY	I	EMAIL	I
+		+-		-+-		+		-+		+		-+
I	2	l	BHAVESH	I	SHARMA	I	24	I	30000	I	BHAVESH@GCOMPANY.IN	I
I	4	l	DEEPIKA	I	GUPTA	I	26	١	55000	١	DEEPIKA@GCOMPANY.IN	I
I	6	l	EKTA	I	YADAV	I	28	١	35000	I	YADAV@GCOMPANY.IN	I
1	8	I	HARSHITA	I	REDDY	I	29	I	56000	I	HARSHITA@GCOMPANY.IN	
I	10		JANU	I	MUKHERJEE	I	30	١	53000	I	JANU@GCOMPANY.IN	I
1	2	I	BHAVESH	I	SHARMA	I	24	I	30000	I	BHAVESH@GCOMPANY.IN	I
1	4	I	DEEPIKA	I	GUPTA	I	26	I	55000	I	DEEPIKA@GCOMPANY.IN	
1	6	l	EKTA	I	YADAV	I	28	I	35000	I	YADAV@GCOMPANY.IN	I
I	8	l	HARSHITA	I	REDDY	I	29	I	56000	١	HARSHITA@GCOMPANY.IN	
I	10	l	JANU	I	MUKHERJEE	I	30	I	53000	١	JANU@GCOMPANY.IN	
I	2	l	BHAVESH	I	SHARMA	I	24	I	30000	١	BHAVESH@GCOMPANY.IN	
1	4	I	DEEPIKA	I	GUPTA	I	26	I	55000	I	DEEPIKA@GCOMPANY.IN	
I	6	l	EKTA	I	YADAV	I	28	I	35000	I	YADAV@GCOMPANY.IN	I
1	8	I	HARSHITA	I	REDDY	I	29	I	56000	I	HARSHITA@GCOMPANY.IN	
I	10	l	JANU	I	MUKHERJEE	I	30	I	53000	١	JANU@GCOMPANY.IN	
I	2	l	BHAVESH	I	SHARMA	I	24	I	30000	I	BHAVESH@GCOMPANY.IN	1
I	4	l	DEEPIKA	I	GUPTA	I	26	I	55000	١	DEEPIKA@GCOMPANY.IN	
I	6	l	EKTA	I	YADAV	I	28	I	35000	I	YADAV@GCOMPANY.IN	1
I	8	l	HARSHITA	I	REDDY	I	29	I	56000	I	HARSHITA@GCOMPANY.IN	I
I	10	l	JANU	I	MUKHERJEE	I	30	I	53000	I	JANU@GCOMPANY.IN	I
+		+-		-+-		+		-+		+		-+

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.

This query will retrieve all the rows with even id values from the EMPLOYEE.

WHAT IF THE TABLE DOESN'T HAVE ID COLUMN

SET @row_number := -1;

SELECT * FROM (SELECT *,@row_number := @row_number + 1 AS
row_num FROM EMPLOYEE) AS numbered_table WHERE row_num % 2
= 0;

+	+-		-+-		-+		+		-+		+-	+
ID	1	FIRST_NAME	I	LAST_NAME	I	AGE	I	SALARY	I	EMAIL	I	row_num
+	+-		-+-		-+		+		-+		+-	+
l	1	ARUN	I	PATEL	I	22	I	40000	I	ARUN@GCOMPANY.IN	I	0
1	3	CHAITANYA	I	SINGH	I	23	I	50000	I	CHAITANYA@GCOMPANY.IN	I	2
1	5	DHANUSH	I	KUMAR	I	25	I	20000	I	DHANUSH@GCOMPANY.IN	I	4
1	7	GAURAV	I	RAO	I	21	I	60000	I	GAURAV@GCOMPANY.IN	I	6
1	9	ISHAAN	I	REDDY	I	32	I	70000	I	ISHAAN@GCOMPANY.IN	I	8
l	1	ARUN	I	PATEL	I	22	I	40000	I	ARUN@GCOMPANY.IN	I	10
1	3	CHAITANYA	I	SINGH	I	23	I	50000	I	CHAITANYA@GCOMPANY.IN	I	12
1	5	DHANUSH	I	KUMAR	I	25	I	20000	I	DHANUSH@GCOMPANY.IN	I	14
1	7	GAURAV	I	RAO	I	21	I	60000	I	GAURAV@GCOMPANY.IN	I	16
1	9	ISHAAN	I	REDDY	I	32	I	70000	I	ISHAAN@GCOMPANY.IN	I	18
1	1	ARUN	I	PATEL	I	22	I	40000	I	ARUN@GCOMPANY.IN	I	20
1	3	CHAITANYA	I	SINGH	I	23	I	50000	I	CHAITANYA@GCOMPANY.IN	I	22
1	5	DHANUSH	I	KUMAR	I	25	I	20000	I	DHANUSH@GCOMPANY.IN	I	24
1	7	GAURAV	I	RAO	I	21	I	60000	I	GAURAV@GCOMPANY.IN	I	26
1	9	ISHAAN	I	REDDY	I	32	I	70000	I	ISHAAN@GCOMPANY.IN	I	28
1	1	ARUN	I	PATEL	I	22	I	40000	I	ARUN@GCOMPANY.IN	I	30
1	3	CHAITANYA	I	SINGH	I	23	Ī	50000	I	CHAITANYA@GCOMPANY.IN	I	32
1	5	DHANUSH	I	KUMAR	I	25	I	20000	I	DHANUSH@GCOMPANY.IN	I	34
I	7	GAURAV	I	RAO	I	21	l	60000	I	GAURAV@GCOMPANY.IN	I	36
I	9	ISHAAN	I	REDDY	I	32	I	70000	I	ISHAAN@GCOMPANY.IN	I	38
+	+-		-+-		-+		+-		-+		.+.	+

Explanation:

SET @row_number := -1;: Initializes a user-defined variable @row_number and sets it to -1. This variable will be used to assign a row number to each record.

SELECT *, @row_number := @row_number + 1 AS row_num FROM my_table: This subquery assigns a row number to each record. The @row_number variable is incremented by 1 for each row, effectively numbering the rows.

AS numbered_table: This gives the subquery a name (numbered_table) that we can reference in the outer query.

WHERE row_num % 2 = 0; This condition selects only the rows where the row number is even. This effectively gives you alternative records.

USING DISTINCT

SELECT DISTINCT ID, FIRST_NAME, LAST_NAME, AGE, EMAIL, SALARY FROM EMPLOYEE;

++												
	ID			FIRST_NAME		LAST_NAME	I	AGE	I	EMAIL	:	SALARY
	+		+-		+-		-+-		+		+-	+
		1	l	ARUN		PATEL	1	22	١	ARUN@GCOMPANY.IN	I	40000
		2	l	BHAVESH		SHARMA	1	24	١	BHAVESH@GCOMPANY.IN	I	30000
		3		CHAITANYA		SINGH	1	23	I	CHAITANYA@GCOMPANY.IN	I	50000
		4		DEEPIKA		GUPTA	1	26	I	DEEPIKA@GCOMPANY.IN		55000
		5		DHANUSH		KUMAR	1	25	I	DHANUSH@GCOMPANY.IN		20000
		6		EKTA		YADAV	1	28	I	YADAV@GCOMPANY.IN	I	35000
		7		GAURAV		RAO	1	21	I	GAURAV@GCOMPANY.IN		60000
		8		HARSHITA		REDDY	1	29	I	HARSHITA@GCOMPANY.IN	I	56000
	I	9	ı	ISHAAN	ı	REDDY	1	32	ı	ISHAAN@GCOMPANY.IN	ı	70000

- If we want to get only unique records then we should use DISTINCT.
- If 2 records has same ID, FIRST_NAME, LAST_NAME, AGE, EMAIL, SALARY in the EMPLOYEE table then only one record will be selected.

USING GROUP BY

10 rows in set (0.00 sec)

SELECT * FROM EMPLOYEE GROUP BY ID, FIRST_NAME, LAST_NAME,
AGE, EMAIL, SALARY;

+ I	:D	+- 	FIRST_NAME	+- 		-	AGE	·+· 	SALARY	·+ 	EMAIL
+		+ -		+-		-+		+		+	
	1		ARUN	I	PATEL	I	22	I	40000	I	ARUN@GCOMPANY.IN
l	2		BHAVESH	I	SHARMA	١	24	I	30000	١	BHAVESH@GCOMPANY.IN
	3	l	CHAITANYA	I	SINGH	I	23	I	50000	I	CHAITANYA@GCOMPANY.IN
	4	l	DEEPIKA	I	GUPTA	I	26	I	55000	I	DEEPIKA@GCOMPANY.IN
	5	l	DHANUSH	I	KUMAR	I	25	I	20000	I	DHANUSH@GCOMPANY.IN
	6	l	EKTA	I	YADAV	١	28	I	35000	١	YADAV@GCOMPANY.IN
	7	l	GAURAV	I	RAO	١	21	I	60000	١	GAURAV@GCOMPANY.IN
	8		HARSHITA	I	REDDY	I	29	I	56000	١	HARSHITA@GCOMPANY.IN
	9		ISHAAN	I	REDDY	I	32	I	70000	I	ISHAAN@GCOMPANY.IN
	10	l	JANU	I	MUKHERJEE	١	30	I	53000	١	JANU@GCOMPANY.IN
. – –		+-		+-		-+		+		+	

- Whichever the records has same ID, FIRST_NAME,
 LAST_NAME, AGE, EMAIL, SALARY in the EMPLOYEE table are grouping into one.
- Totally 10 groups are creating, each group contains 4 records of same data.
- From every group only one record is displayed.
- DISTINCT and GROUP BY are similar.
- MySQL does not have a built-in concept of a "ROWID" like some other databases (e.g., Oracle) do. In MySQL, you typically rely on primary keys (which are unique) to uniquely identify rows in a table.

USE OF CONSTRAINTS

DROP TABLE IF EXISTS WORKTAB1;

CREATE TABLE WORKTAB1(ID INTEGER, NAME VARCHAR2(90), AGE INTEGER);

INSERT INTO WORKTAB1(ID) VALUES(1);

SELECT * FROM WORKTAB1;

+----+

ID | NAME | AGE |

+----+

NULL | NULL |

+----+

1 row in set (0.00 sec)

By default column allows NULL values.

```
FSD Training Program
```

```
INSERT INTO WORKTAB1(ID, NAME) VALUES(2, 'ABC');
SELECT * FROM WORKTAB1;
+----+
+----+
| 1 | NULL | NULL |
| 2 | ABC | NULL |
+----+
2 rows in set (0.00 sec)
INSERT INTO WORKTAB1(ID, AGE) VALUES(3, 33);
Query OK, 1 row affected (0.01 sec)
SELECT * FROM WORKTAB1;
+----+
+----+
   1 | NULL | NULL |
 2 | ABC | NULL |
   3 | NULL | 33 |
+----+
3 rows in set (0.00 sec)
```

```
INSERT INTO WORKTAB1(NAME, AGE) VALUES('AMAN', 23);
Query OK, 1 row affected (0.01 sec)
SELECT * FROM WORKTAB1;
+----+
+----+
   1 | NULL | NULL |
 2 | ABC | NULL |
   3 | NULL | 33 |
+----+
4 rows in set (0.00 sec)
INSERT INTO WORKTAB1(NAME) VALUES('MANOHAR');
Query OK, 1 row affected (0.01 sec)
SELECT * FROM WORKTAB1;
+----+
+----+
   1 | NULL | NULL |
   2 | ABC | NULL |
   3 | NULL | 33 |
| NULL | MANOHAR | NULL |
+-----+
5 rows in set (0.00 sec)
```

```
FSD Training Program
```

DROP TABLE IF EXISTS WORKTAB2;

Query OK, 0 rows affected, 1 warning (0.01 sec)

CREATE TABLE WORKTAB2(ID INTEGER, NAME VARCHAR(90) NOT NULL, AGE INTEGER);

Query OK, 0 rows affected (0.03 sec)

DESC used to provide information about the structure of the table

DESC WORKTAB2;

+		+		+-		+	+		+	-+
•	Field		Туре				•	Default	•	
 	ID NAME AGE	 	int varchar(90) int	 	YES NO YES	 	 	NULL NULL NULL	 	
+		+		+		+	+		+	+

3 rows in set (0.01 sec)

- By using NOT NULL we can make sure that column are not having null values.
- In one table any number of columns can be NOT NULL.

INSERT INTO WORKTAB2(ID) VALUES(1);

ERROR 1364 (HY000): Field 'NAME' doesn't have a default value

INSERT INTO WORKTAB2(ID, NAME) VALUES(2, 'AMAN');
Query OK, 1 row affected (0.01 sec)

```
FSD Training Program
SELECT * FROM WORKTAB2;
+----+
+----+
   2 | AMAN | NULL |
+----+
1 row in set (0.00 sec)
INSERT INTO WORKTAB2(ID, AGE) VALUES(3, 33);
ERROR 1364 (HY000): Field 'NAME' doesn't have a default
value
INSERT INTO WORKTAB2(NAME, AGE) VALUES('MADHU', 23);
Query OK, 1 row affected (0.01 sec)
SELECT * FROM WORKTAB2;
+----+
+----+
   2 | AMAN | NULL |
```

+-----+

```
FSD Training Program
```

```
INSERT INTO WORKTAB2(NAME) VALUES('MANOHAR');
Query OK, 1 row affected (0.01 sec)
SELECT * FROM WORKTAB2;
+-----+
+-----+
| 2 | AMAN | NULL |
NULL | MADHU |
               23
| NULL | MANOHAR | NULL |
+-----+
3 rows in set (0.00 sec)
INSERT INTO WORKTAB2(AGE) VALUES(25);
ERROR 1364 (HY000): Field 'NAME' doesn't have a default
value
SELECT * FROM WORKTAB2;
+----+
+----+
   2 | AMAN | NULL |
NULL | MADHU |
               23
| NULL | MANOHAR | NULL |
+-----+
3 rows in set (0.00 sec)
```

+----+

```
INSERT INTO WORKTAB3(ID, AGE) VALUES(3, 33);
ERROR 1364 (HY000): Field 'NAME' doesn't have a default
value
INSERT INTO WORKTAB3(NAME, AGE) VALUES('MADHU', 23);
ERROR 1364 (HY000): Field 'ID' doesn't have a default value
INSERT INTO WORKTAB3(NAME) VALUES('MADHU');
ERROR 1364 (HY000): Field 'ID' doesn't have a default value
INSERT INTO WORKTAB3(AGE) VALUES(25);
ERROR 1364 (HY000): Field 'ID' doesn't have a default value
SELECT * FROM WORKTAB3;
+----+
+----+
  2 | AMAN | NULL |
+----+
1 row in set (0.00 sec)
```

```
DROP TABLE WORKTAB4;
ERROR 1051 (42S02): Unknown table 'mysql notes.worktab4'
DROP TABLE IF EXISTS WORKTAB4;
Query OK, 0 rows affected, 1 warning (0.01 sec)
CREATE TABLE WORKTAB4(ID INTEGER, NAME VARCHAR(90), AGE
INTEGER);
Query OK, 0 rows affected (0.03 sec)
SELECT * FROM WORKTAB4;
Empty set (0.00 sec)
DESC WORKTAB4;
+----+
+----+
| NAME | varchar(90) | YES | | NULL |
+----+
```

FSD Training Program

FSD Training Program

```
INSERT INTO WORKTAB4(ID, NAME) VALUES(1, 'MANOHAR');
INSERT INTO WORKTAB4(ID, NAME) VALUES(1, 'MANOHAR');
INSERT INTO WORKTAB4(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
INSERT INTO WORKTAB4(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
INSERT INTO WORKTAB4(NAME, AGE) VALUES('EHTESHAM', 22);
INSERT INTO WORKTAB4(NAME, AGE) VALUES('EHTESHAM', 22);
SELECT * FROM WORKTAB4;
| ID | NAME | AGE |
+----+
    1 | MANOHAR | NULL |
    1 | MANOHAR | NULL |
    2 | AMAN | 22 |
    2 | AMAN | 22 |
| NULL | EHTESHAM | 22 |
 NULL | EHTESHAM | 22 |
+-----+
```

DROP TABLE IF EXISTS WORKTAB5;

CREATE TABLE WORKTAB5(ID INTEGER, NAME VARCHAR(90) UNIQUE, AGE INTEGER);

DESC WORKTAB5;

++		-+		+		+		+-	 +
Field		•		-		•	Default		•
++		-+		-+		+		.+-	 +
ID	int		YES				NULL		
NAME	varchar(90)		YES		UNI	I	NULL		
AGE	int	١	YES			I	NULL	I	
++		-+		-+		+		+-	 +
2 nous in	sot (0.00 s		`						

- 3 rows in set (0.00 sec)
 - By default columns allow duplicate values.
 - In one table any number of columns can be UNIQUE
 - By using UNIQUE we can avoid duplicate values in the same column in the table.
 - UNIQUE column allows any number of NULL values but not duplicate values.
 - Two NULL values are not same i.e. they are not duplicate.

INSERT INTO WORKTAB5(ID, NAME) VALUES(1, 'MANOHAR');
Query OK, 1 row affected (0.01 sec)

INSERT INTO WORKTAB5(ID, NAME) VALUES(1, 'MANOHAR');
ERROR 1062 (23000): Duplicate entry 'MANOHAR' for key
'worktab5.NAME'

```
FSD Training Program
```

```
INSERT INTO WORKTAB5(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
Query OK, 1 row affected (0.00 sec)
INSERT INTO WORKTAB5(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
ERROR 1062 (23000): Duplicate entry 'AMAN' for key
'worktab5.NAME'
INSERT INTO WORKTAB5(NAME, AGE) VALUES('AMAN', 22);
ERROR 1062 (23000): Duplicate entry 'AMAN' for key
'worktab5.NAME'
INSERT INTO WORKTAB5(NAME, AGE) VALUES('EHSTESHAM', 22);
Query OK, 1 row affected (0.00 sec)
INSERT INTO WORKTAB5(ID, AGE) VALUES(3, 22);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB5(ID, AGE) VALUES(3, 22);
Query OK, 1 row affected (0.00 sec)
SELECT * FROM WORKTAB5;
+----+
                 | AGE |
    I NAME
 ID
+----+
    1 | MANOHAR | NULL |
    2 AMAN
            22 |
| NULL | EHSTESHAM |
                    22 |
    3 | NULL |
                    22 |
                    22
    3 | NULL
+-----+
5 rows in set (0.00 sec)
```

```
FSD Training Program
DROP TABLE IF EXISTS WORKTAB6;
Query OK, 0 rows affected, 1 warning (0.00 sec)
CREATE TABLE WORKTAB6(ID INTEGER, NAME VARCHAR(90) UNIQUE,
AGE INTEGER UNIQUE);
Query OK, 0 rows affected (0.03 sec)
DESC WORKTAB6;
+----+
+----+
| NAME | varchar(90) | YES | UNI | NULL |
              | YES | UNI | NULL |
| AGE | int
+----+
3 rows in set (0.00 sec)
INSERT INTO WORKTAB6(ID, NAME) VALUES(1, 'MANOHAR');
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB6(ID, NAME) VALUES(1, 'MANOHAR');
```

ERROR 1062 (23000): Duplicate entry 'MANOHAR' for key

Query OK, 1 row affected (0.00 sec)

INSERT INTO WORKTAB6(ID, NAME, AGE) VALUES(2, 'AMAN', 22);

'worktab6.NAME'

```
FSD Training Program
```

```
INSERT INTO WORKTAB6(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
ERROR 1062 (23000): Duplicate entry 'AMAN' for key
'worktab6.NAME'
INSERT INTO WORKTAB6(NAME, AGE) VALUES('AMAN', 22);
ERROR 1062 (23000): Duplicate entry 'AMAN' for key
'worktab6.NAME'
INSERT INTO WORKTAB6(ID, AGE) VALUES(5, 22);
ERROR 1062 (23000): Duplicate entry '22' for key
'worktab6.AGE'
INSERT INTO WORKTAB6(ID) VALUES(5);
Query OK, 1 row affected (0.00 sec)
INSERT INTO WORKTAB6(ID) VALUES(5);
Query OK, 1 row affected (0.01 sec)
SELECT * FROM WORKTAB6;
+----+
l ID
    NAME
               I AGE
+-----+
    1 | MANOHAR | NULL |
    2 | AMAN
            22 |
    5 | NULL | NULL |
    5 | NULL | NULL |
+-----+
```

```
DROP TABLE IF EXISTS WORKTAB7;
Query OK, 0 rows affected, 1 warning (0.01 sec)
```

CREATE TABLE WORKTAB7(ID INTEGER, NAME VARCHAR(90), AGE INTEGER, CONSTRAINT WORKTAB7_UK1 UNIQUE(NAME), CONSTRAINT WORKTAB7_UK2 UNIQUE(AGE));

Query OK, 0 rows affected (0.03 sec)

- Syntax CONSTRAINT(declaration) WORKTAB7_UK1(IDENTIFIER)
 UNIQUE(NAME)(type of the constraint and column name)
- Every constraints should be having unique identifier names in across the tables.
- We can disable or permanently drop the constraints. It is the better approach than previous.

```
INSERT INTO WORKTAB7(ID, NAME) VALUES(1, 'MANOHAR');
Query OK, 1 row affected (0.01 sec)
```

INSERT INTO WORKTAB7(ID, NAME) VALUES(1, 'MANOHAR');
ERROR 1062 (23000): Duplicate entry 'MANOHAR' for key
'worktab7.WORKTAB7_UK1'

INSERT INTO WORKTAB7(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
Query OK, 1 row affected (0.00 sec)

INSERT INTO WORKTAB7(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
ERROR 1062 (23000): Duplicate entry 'AMAN' for key
'worktab7.WORKTAB7_UK1'

```
FSD Training Program
```

```
INSERT INTO WORKTAB7(NAME, AGE) VALUES('EHSTESHAM', 22);
ERROR 1062 (23000): Duplicate entry '22' for key
'worktab7.WORKTAB7 UK2'
INSERT INTO WORKTAB7(ID, AGE) VALUES(5, 22);
ERROR 1062 (23000): Duplicate entry '22' for key
'worktab7.WORKTAB7 UK2'
INSERT INTO WORKTAB7(ID) VALUES(5);
Query OK, 1 row affected (0.03 sec)
INSERT INTO WORKTAB7(ID) VALUES(6);
Query OK, 1 row affected (0.00 sec)
SELECT * FROM WORKTAB7;
+----+
+----+
    1 | MANOHAR | NULL |
    2 AMAN
                  22 |
    5 | NULL | NULL |
    6 | NULL | NULL |
+----+
4 rows in set (0.00 sec)
```

```
FSD Training Program
DROP TABLE IF EXISTS WORKTAB8;
Query OK, 0 rows affected, 1 warning (0.01 sec)
CREATE TABLE WORKTAB8(ID INTEGER, NAME VARCHAR(90), AGE
INTEGER, CONSTRAINT WORKTAB8 UK1 UNIQUE(NAME, AGE));
Query OK, 0 rows affected (0.02 sec)
INSERT INTO WORKTAB8(ID, NAME) VALUES(1, 'AMAN');
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID, NAME) VALUES(1, 'AMAN');
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID, NAME) VALUES(1, 'AMAN');
Query OK, 1 row affected (0.00 sec)
DESC WORKTAB8;
+----+
+----+
NAME | varchar(90) | YES | MUL | NULL |
```

+----+

FSD Training Program

- In the above constraint two records cant be having same values for the NAME and AGE columns.
- We can refer to it as a composite unique key.

```
INSERT INTO WORKTAB8(ID, NAME, AGE) VALUES(1, 'AMAN', 22);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID, NAME, AGE) VALUES(1, 'AMAN', 22);
ERROR 1062 (23000): Duplicate entry 'AMAN-22' for key
'worktab8.WORKTAB8 UK1'
INSERT INTO WORKTAB8(ID, NAME, AGE) VALUES(1, 'AMAN', 23);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID, NAME, AGE) VALUES(1, 'AMAN', 23);
ERROR 1062 (23000): Duplicate entry 'AMAN-23' for key
'worktab8.WORKTAB8 UK1'
INSERT INTO WORKTAB8(ID, NAME, AGE) VALUES(1, 'AMAN', 24);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID, NAME, AGE) VALUES(1, 'MANOHAR',
25);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID, NAME, AGE) VALUES(1, 'EHTESHAM',
25);
Query OK, 1 row affected (0.00 sec)
INSERT INTO WORKTAB8(ID) VALUES(6);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID) VALUES(6);
Query OK, 1 row affected (0.00 sec)
```

• We don't get error because NULL, NULL is not considered as combination.

```
INSERT INTO WORKTAB8(ID, NAME) VALUES(1, 'AMAN');
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB8(ID, NAME) VALUES(1, 'AMAN');
Query OK, 1 row affected (0.00 sec)
```

- We don't get error because AMAN, NULL and again AMAN,
 NULL is not considered as combination.
- The reason you're not seeing an error when inserting a record with a value and NULL in the same columns is because NULL is considered a distinct value. Therefore, a combination of ('John', NULL) and ('John', NULL) is considered unique according to the rules of MySQL

```
INSERT INTO TAB8(ID, NAME, AGE) VALUES(2, 'RAMU', 22);
INSERT INTO TAB8(ID, NAME, AGE) VALUES(2, 'RAMU', 23);
INSERT INTO TAB8(ID, NAME, AGE) VALUES(2, 'AMU', 23);
INSERT INTO TAB8(ID, NAME, AGE) VALUES(2, 'RAMU', 22);//ERROR
INSERT INTO TAB8(NAME, AGE) VALUES('RAMU', 22);//ERROR
INSERT INTO TAB8(NAME, AGE) VALUES('RAMU', 22);//ERROR
INSERT INTO TAB8(ID, AGE) VALUES(5, 22); //ERROR
INSERT INTO TAB8(ID, AGE) VALUES(5, 22); //ERROR
INSERT INTO TAB8(ID) VALUES(5);
INSERT INTO TAB8(ID) VALUES(5);
INSERT INTO TAB8(ID) VALUES(5);
INSERT INTO TAB8(ID) VALUES(5);
INSERT INTO TAB8(ID) VALUES(5);
```

• The above queries possible because under UNIQUE constraint any number of NULL values and that won't be considered as a combination.

```
FSD Training Program
INSERT INTO TAB8(AGE) VALUES(25);
SELECT * FROM TAB8;
DROP TABLE IF EXISTS WORKTAB9;
Query OK, 0 rows affected, 1 warning (0.05 sec)
CREATE TABLE WORKTAB9(ID INTEGER, NAME VARCHAR(90) PRIMARY
KEY, AGE INTEGER);
Query OK, 0 rows affected (0.07 sec)
  • PRIMARY KEY is a combination of NOT NULL and UNIQUE.
  • If any column is declared as PRIMARY KEY then that
    column value should not be NULL value and should not
    contain duplicate value.
INSERT INTO WORKTAB9(ID, NAME) VALUES(1, 'MANOHAR');
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB9(ID, NAME) VALUES(1, 'MANOHAR');
ERROR 1062 (23000): Duplicate entry 'MANOHAR' for key
'worktab9.PRIMARY'
INSERT INTO WORKTAB9(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB9(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
```

ERROR 1062 (23000): Duplicate entry 'AMAN' for key

INSERT INTO WORKTAB9(ID, AGE) VALUES(5, 22);

'worktab9.PRIMARY'

```
FSD Training Program
ERROR 1364 (HY000): Field 'NAME' doesn't have a default
value
INSERT INTO WORKTAB9(ID) VALUES(5);
ERROR 1364 (HY000): Field 'NAME' doesn't have a default
value
INSERT INTO WORKTAB9(NAME) VALUES('VIJAY');
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB9(NAME) VALUES('VIJAY');
ERROR 1062 (23000): Duplicate entry 'VIJAY' for key
'worktab9.PRIMARY'
SELECT * FROM WORKTAB9;
+----+
| ID | NAME
                 l AGE l
+----+
    2 | AMAN | 22 |
NULL | EHSTESHAM | 22 |
```

1 | MANOHAR | NULL |

| NULL | VIJAY | NULL |

+----+

```
FSD Training Program
```

CREATE TABLE WORKTAB10(ID INTEGER, NAME VARCHAR(90) PRIMARY KEY, AGE INTEGER PRIMARY KEY);

ERROR 1068 (42000): Multiple primary key defined

• In a table there should be only one column declared as PRIMARY KEY not more than one column.

```
CREATE TABLE WORKTAB11(ID INTEGER, NAME VARCHAR(90), AGE
INTEGER, CONSTRAINT WORKTAB11 PK1 PRIMARY KEY(AGE));
Query OK, 0 rows affected (0.04 sec)
INSERT INTO WORKTAB11(ID, NAME) VALUES(1, 'MANOHAR');
ERROR 1364 (HY000): Field 'AGE' doesn't have a default
value
INSERT INTO WORKTAB11(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
Query OK, 1 row affected (0.00 sec)
INSERT INTO WORKTAB11(ID, NAME, AGE) VALUES(2, 'AMAN', 22);
ERROR 1062 (23000): Duplicate entry '22' for key
'worktab11.PRIMARY'
INSERT INTO WORKTAB11(NAME, AGE) VALUES('AMAN', 22);
ERROR 1062 (23000): Duplicate entry '22' for key
'worktab11.PRIMARY'
INSERT INTO WORKTAB11(NAME, AGE) VALUES('AMAN', 28);
Query OK, 1 row affected (0.00 sec)
INSERT INTO WORKTAB11(ID, AGE) VALUES(5, 28);
ERROR 1062 (23000): Duplicate entry '28' for key
'worktab11.PRIMARY'
INSERT INTO WORKTAB11(ID) VALUES(5);
```

```
ERROR 1364 (HY000): Field 'AGE' doesn't have a default
value
INSERT INTO WORKTAB11(AGE) VALUES(25);
Query OK, 1 row affected (0.00 sec)
INSERT INTO WORKTAB11(NAME) VALUES('VIJAY');
ERROR 1364 (HY000): Field 'AGE' doesn't have a default
value
SELECT * FROM WORKTAB11;
+----+
+----+
    2 | AMAN | 22 |
+----+
CREATE TABLE WORKTAB12(ID INTEGER, NAME VARCHAR(90), AGE
INTEGER, CONSTRAINT WORKTAB12 PK1 PRIMARY KEY(AGE, NAME));
 • Composite PRIMARY key is possible.
INSERT INTO WORKTAB12 VALUES(1, 'AMAN', 22);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB12 VALUES(2, 'AMAN', 22);
ERROR 1062 (23000): Duplicate entry '22-AMAN' for key
'worktab12.PRIMARY'
INSERT INTO WORKTAB12 VALUES(3, 'AMAN', 23);
Query OK, 1 row affected (0.01 sec)
INSERT INTO WORKTAB12 VALUES(4, 'MANOHAR', 23);
Query OK, 1 row affected (0.00 sec)
```

```
INSERT INTO WORKTAB12(ID, NAME) VALUES(5, 'JAGAN');
ERROR 1364 (HY000): Field 'AGE' doesn't have a default
value
INSERT INTO WORKTAB12(ID, AGE) VALUES(6, 25);
ERROR 1364 (HY000): Field 'NAME' doesn't have a default
value
```

FOREIGN KEY

CREATE TABLE FS_STUDENT(ID INTEGER UNIQUE, FIRST_NAME VARCHAR(90), LAST_NAME VARCHAR(90), AGE INTEGER, EMAIL VARCHAR(90));

Query OK, 0 rows affected (0.04 sec)

CREATE TABLE FS_STUDENT_ADDRESS(HOUSE_NO VARCHAR(90), STREET_NAME VARCHAR(90), CITY VARCHAR(90), STATE VARCHAR(90), FS_STUDENT_ID INTEGER, CONSTRAINT FS_STUDENT_ADDRESS_FK1 FOREIGN KEY(FS_STUDENT_ID) REFERENCES FS_STUDENT(ID));

Query OK, 0 rows affected (0.03 sec)

• For FOREIGN KEY purpose REFERENCES table column either UNIQUE or PRIMARY

```
FSD Training Program
INSERT INTO FS STUDENT VALUES(1, 'AMAN', 'GUPTA', 22,
'AMAN@VP.COM');
Query OK, 1 row affected (0.01 sec)
INSERT INTO FS STUDENT ADDRESS VALUES('123/A', 'BTM',
'BLR', 'KAR', 1);
Query OK, 1 row affected (0.01 sec)
INSERT INTO FS_STUDENT VALUES(2, 'MANOHAR', 'VERMA', 24,
'MANOHAR@VP.COM');
Query OK, 1 row affected (0.01 sec)
INSERT INTO FS STUDENT ADDRESS VALUES('256/B', 'BSK',
'BLR', 'KAR', 2);
Query OK, 1 row affected (0.00 sec)
INSERT INTO FS STUDENT VALUES(3, 'VIJAY', 'KUMAR', 26,
'VIJAY@VP.COM');
Query OK, 1 row affected (0.00 sec)
```

INSERT INTO FS STUDENT ADDRESS VALUES('126/C', 'JPN',

'BLR', 'KAR', 3);

Query OK, 1 row affected (0.01 sec)

```
INSERT INTO FS_STUDENT_ADDRESS VALUES('450/D', 'KKC',
'BLR', 'KAR', 5);

ERROR 1452 (23000): Cannot add or update a child row: a
foreign key constraint fails
(`mysql_notes`.`fs_student_address`, CONSTRAINT
`FS_STUDENT_ADDRESS_FK1` FOREIGN KEY (`FS_STUDENT_ID`)
REFERENCES `fs_student` (`ID`))
```

- Trying to insert child record straight away without parent.
- FOREIGN KEY should have reference value of the column from the parent.

```
INSERT INTO FS_STUDENT VALUES(4, 'JAGAN', 'REDDY', 27,
'JAGAN@VP.COM');
Query OK, 1 row affected (0.01 sec)

INSERT INTO FS_STUDENT_ADDRESS VALUES('450/D', 'KKC',
'BLR', 'KAR', 4);
Query OK, 1 row affected (0.01 sec)

DELETE FROM FS_STUDENT WHERE ID = 4;
ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails
(`mysql_notes`.`fs_student_address`, CONSTRAINT
`FS_STUDENT_ADDRESS_FK1` FOREIGN KEY (`FS_STUDENT_ID`)
REFERENCES `fs_student` (`ID`))
```

- FS_STUDENT ID = 4 having child in the FS_STUDENT_ADDRESS.
- You can't delete parent record without deleting a child record.

```
DELETE FROM FS_STUDENT_ADDRESS WHERE FS_STUDENT_ID = 4;

Query OK, 1 row affected (0.01 sec)

DELETE FROM FS_STUDENT WHERE ID = 4;

Query OK, 1 row affected (0.01 sec)

DROP TABLE FS_STUDENT;

ERROR 3730 (HY000): Cannot drop table 'fs_student' referenced by a foreign key constraint 'FS_STUDENT_ADDRESS_FK1' on table 'fs_student_address'.

DROP TABLE FS_STUDENT_ADDRESS;

Query OK, 0 rows affected (0.02 sec)

DROP TABLE FS_STUDENT;
```

DROP TABLE FS_STUDENT;
Query OK, 0 rows affected (0.01 sec)

- Straight away we cant delete STUDENT table.
- STUDENT table is a parent to ADDRESS table
- Without deleting the child we cant able to delete PARENT
- Even though ADDRESS table is empty we cant able to drop STUDENT table.
- First we need to drop ADDRESS table then only we can able to drop the STUDENT table.

CREATE TABLE FS_STUDENT(ID INTEGER, FIRST_NAME VARCHAR(90), LAST_NAME VARCHAR(90), AGE INTEGER, EMAIL VARCHAR(90));

Ouerv OK, 0 rows affected (0.03 sec)

CREATE TABLE FS_STUDENT_ADDRESS(HOUSE_NO VARCHAR(90), STREET_NAME VARCHAR(90), CITY VARCHAR(90), STATE VARCHAR(90), FS_STUDENT_ID INTEGER, CONSTRAINT FS_STUDENT_ADDRESS_FK1 FOREIGN KEY(FS_STUDENT_ID) REFERENCES FS_STUDENT(ID));

ERROR 1822 (HY000): Failed to add the foreign key
constraint. Missing index for constraint
'FS_STUDENT_ADDRESS_FK1' in the referenced table
'fs student'

DROP TABLE FS STUDENT;

CREATE TABLE FS_STUDENT(ID INTEGER UNIQUE, FIRST_NAME VARCHAR(90), LAST_NAME VARCHAR(90), AGE INTEGER, EMAIL VARCHAR(90));

Query OK, 0 rows affected (0.04 sec)

CREATE TABLE FS_STUDENT_ADDRESS(HOUSE_NO VARCHAR(90), STREET_NAME VARCHAR(90), CITY VARCHAR(90), STATE VARCHAR(90), FS_STUDENT_ID INTEGER, CONSTRAINT FS_STUDENT_ADDRESS_FK1 FOREIGN KEY(FS_STUDENT_ID) REFERENCES FS_STUDENT(ID));

```
INSERT INTO FS_STUDENT VALUES(1, 'AMAN', 'GUPTA', 22,
'AMAN@GMAIL.COM');
Query OK, 1 row affected (0.01 sec)
```

INSERT INTO FS_STUDENT_ADDRESS(HOUSE_NO, STREET_NAME, CITY,
STATE) VALUES('140/F', 'RRN', 'BLR', 'KAR');

Query OK, 1 row affected (0.01 sec)

- we can have NULL values for FOREIGN KEY REFERENCE.
- If FS_STUDENT_ID is not PRIMARY KEY in the ADDRESS we can have NULL values.
- We are inserting an ADDRESS which doesn't belong to any STUDENT.
- By default FOREIGN KEY allows NULL values.

```
UPDATE FS_STUDENT_ADDRESS SET FS_STUDENT_ID = 3 WHERE
HOUSE_NO = '140/F';
ERROR 1452 (23000): Cannot add or update a child row: a
foreign key constraint fails
(`mysql_notes`.`fs_student_address`, CONSTRAINT
`FS_STUDENT_ADDRESS_FK1` FOREIGN KEY (`FS_STUDENT_ID`)
REFERENCES `fs_student` (`ID`))
```

• There is no corresponding record.

```
UPDATE FS_STUDENT_ADDRESS SET FS_STUDENT_ID = 1 WHERE
HOUSE_NO = '140/F';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

```
INSERT INTO FS_STUDENT(FIRST_NAME, LAST_NAME, AGE, EMAIL)
VALUES('MANOHAR', 'VERMA', 22, 'MANOHAR@GMAIL.COM');
Ouerv OK, 1 row affected (0.03 sec)
```

INSERT INTO FS_STUDENT_ADDRESS(HOUSE_NO, STREET_NAME, CITY,
STATE) VALUES('224/Y', 'RGN', 'BLR', 'KAR');
Query OK, 1 row affected (0.01 sec)

- In the base table ID column is NULL and in the child table FS_STUDENT_ID is also NULL.
- In the parent table there is a record with no ID.
- In the child table there is record which doesn't match to any of the parent table records.
- NULL cant be assigned to another NULL.
- NULL cant be mapped to another NULL.

```
UPDATE FS_STUDENT SET ID = 2 WHERE FIRST_NAME = 'MANOHAR';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

UPDATE FS_STUDENT_ADDRESS SET FS_STUDENT_ID = 2 WHERE
HOUSE_NO = '224/Y';

Query OK, 1 row affected (0.01 sec)

Rows matched: 1 Changed: 1 Warnings: 0

ONE-TO-ONE

DROP TABLE IF EXISTS PERSON;

CREATE TABLE PERSON (ID INTEGER PRIMARY KEY, FIRST_NAME VARCHAR(90), AGE INTEGER);

• Because of ID column is PRIMARY KEY PERSON table can become a parent to child table.

DROP TABLE IF EXISTS ADDRESS;

CREATE TABLE ADDRESS(HOUSE_NO VARCHAR(90), STREET_NAME VARCHAR(90), CITY VARCHAR(90), STATE VARCHAR(90), PERSON_ID INTEGER UNIQUE, CONSTRAINT ADDRESS_FK1 FOREIGN KEY(PERSON_ID) REFERENCES PERSON(ID));

```
INSERT INTO ADDRESS VALUES('185/A', 'BSK', 'BLR', 'KAR',
1);
Query OK, 1 row affected (0.01 sec)
INSERT INTO ADDRESS VALUES('224/B', 'JPN', 'BLR', 'KAR',
1);
ERROR 1062 (23000): Duplicate entry '1' for key
'address.PERSON ID'
  • FOREIGN KEY is a UNIQUE so we can't insert duplicates.
  • One record of PERSON mapping to only one record of
    ADDRESS. So we call it as one to one mapping.
INSERT INTO ADDRESS VALUES('185/A', 'BSK', 'BLR', 'KAR',
1);
Query OK, 1 row affected (0.01 sec)
INSERT INTO ADDRESS VALUES('224/B', 'JPN', 'BLR', 'KAR',
1);
ERROR 1062 (23000): Duplicate entry '1' for key
'address.PERSON_ID'
INSERT INTO ADDRESS VALUES('228/C', 'RRN', 'BLR', 'KAR',
2);
Query OK, 1 row affected (0.01 sec)
INSERT INTO ADDRESS VALUES('356/D', 'KKC', 'BLR', 'KAR',
3);
Query OK, 1 row affected (0.00 sec)
```

```
FSD Training Program
```

```
INSERT INTO ADDRESS VALUES('521/F', 'RJN', 'BLR', 'KAR',
4);
```

Query OK, 1 row affected (0.01 sec)

INSERT INTO ADDRESS VALUES('652/G', 'KRL', 'BLR', 'KAR',
6);

ERROR 1452 (23000): Cannot add or update a child row: a
foreign key constraint fails (`mysql_notes`.`address`,
CONSTRAINT `ADDRESS_FK1` FOREIGN KEY (`PERSON_ID`)
REFERENCES `person` (`ID`))

No PERSON with ID as 6

INSERT INTO ADDRESS(HOUSE_NO, STREET_NAME, CITY, STATE)
VALUES('224/I', 'BTM', 'BLR', 'KAR');

• We can insert ADDRESS without choosing PERSON_ID because this column is UNIQUE and allows NULL values.

SELECT * FROM PERSON;

+		+		+		+		+
I	ID	1	FIRST_NAME		LAST_NAME	I	AGE	I
+		+		+		+		+
1	1		AMAN		GUPTA	I	22	I
	2		MANOHAR		VERMA	I	24	I
1	3		SWETHA		SHARMA		21	I
	4		VIJAY		VAISHNAV	I	23	I
	5		KUMAR		SINHA	I	23	I
+		+		+		+-		+

```
SELECT * FROM ADDRESS;
+----+
| HOUSE_NO | STREET_NAME | CITY | STATE | PERSON_ID |
+----+
185/A | BSK | BLR | KAR |
                      1 I
2 I
3 I
4 I
           BLR KAR NULL
| 224/I | BTM
+----+
SELECT * FROM PERSON WHERE FIRST_NAME = 'AMAN';
+---+------
| ID | FIRST_NAME | LAST_NAME | AGE |
+---+
 1 | AMAN | GUPTA | 22 |
SELECT * FROM ADDRESS WHERE HOUSE NO = '185/A';
+----+
| HOUSE NO | STREET NAME | CITY | STATE | PERSON ID |
+----+
```

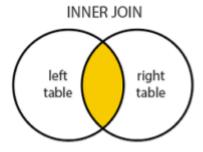
+----+

```
SELECT * FROM ADDRESS WHERE PERSON ID = 3;
+----+
| HOUSE NO | STREET NAME | CITY | STATE | PERSON ID |
+----+
| 356/D | KKC | BLR | KAR |
+----+
SELECT * FROM ADDRESS WHERE PERSON ID = (SELECT ID FROM
PERSON WHERE FIRST_NAME = 'AMAN');
+----+
| HOUSE_NO | STREET_NAME | CITY | STATE | PERSON_ID |
+----+
+----+
SELECT * FROM PERSON WHERE ID = (SELECT PERSON ID FROM
ADDRESS WHERE HOUSE_NO = '185/A');
+---+
| ID | FIRST_NAME | LAST_NAME | AGE |
+---+
+---+
```

FSD Training Program

INNER JOIN

 The default join is inner join, if you are not specifying any keywords.



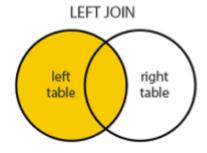
SELECT * FROM PERSON, ADDRESS WHERE PERSON.ID = ADDRESS.PERSON_ID;

I	ID	I	FIRST_NAME		LAST_NAME	A	IGE	I	HOUSE_NO	I	STREET_NAME	I	CITY	I	STATE	I	PERSON_ID
			AMAN								BSK				KAR		
I	2	I	MANOHAR	I	VERMA	I	24	I	228/C	I	RRN	I	BLR	I	KAR	I	2
I	3	I	SWETHA	I	SHARMA	I	21	I	356/D	I	ККС	I	BLR	I	KAR	I	3
I	4	I	VIJAY	I	VAISHNAV	I	23	I	521/F	I	RJN	I	BLR	I	KAR	I	4
+		-+-		+		+		+		-+		+		-+		.+	+

SELECT * FROM PERSON P, ADDRESS A WHERE P.ID = A.PERSON_ID;

	SELECT * FROM PERSON P INNER JOIN ADDRESS A ON P.ID = A.PERSON_ID;																
ID FIRST_NAME LAST_NAME AGE HOUSE_NO STREET_NAME CITY STATE PERSON_ID															+		
ı	ID	I	FIRST_NAME	I	LAST_NAME	I	AGE	I	HOUSE_NO	ı	STREET_NAME	ı	CITY	I	STATE	PERSON_	_ID
+-		+-		+		+-		+		+		+		+		+	+
Ī	1	I	AMAN	I	GUPTA	I	22	I	185/A	I	BSK	I	BLR	I	KAR	I	1
I	2	I	MANOHAR	I	VERMA	I	24	I	228/C	I	RRN	١	BLR	I	KAR	I	2
I	3	I	SWETHA	I	SHARMA	I	21	I	356/D	I	ККС	I	BLR	I	KAR	l	3
I	4	I	VIJAY	١	VAISHNAV	I	23	١	521/F	I	RJN	١	BLR	I	KAR	I	4
+-		+-		+		+-		+		+		+		+		+	+

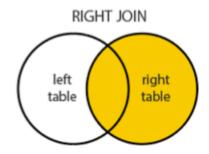
LEFT OUTER JOIN



SELECT * FROM PERSON P LEFT OUTER JOIN ADDRESS A ON P.ID = A.PERSON_ID;

+	+		+		+-		+		+		+		+		+-	+
1	D	FIRST_NAME	I	LAST_NAME		AGE	I	HOUSE_NO	I	STREET_NAME	I	CITY	I	STATE	I	PERSON_ID
+	+		+		+-		+		+		+		+		+-	+
I	1	AMAN	I	GUPTA	I	22	I	185/A	I	BSK	١	BLR	I	KAR		1
	2	MANOHAR	I	VERMA		24	I	228/C	I	RRN	I	BLR	I	KAR	I	2
	3	SWETHA	١	SHARMA		21	١	356/D	I	KKC	I	BLR	I	KAR	I	3
	4	VIJAY	١	VAISHNAV		23	١	521/F	I	RJN	I	BLR	I	KAR	I	4
	5	KUMAR	١	SINHA		23	١	NULL	I	NULL	I	NULL	I	NULL	I	NULL
+	+		- +		+-		+		. +		. +		. +		+-	+

RIGHT OUTER JOIN



SELECT * FROM PERSON P RIGHT OUTER JOIN ADDRESS A ON P.ID =
A.PERSON ID;

SELECT * FROM PERSON P RIGHT OUTER JOIN ADDRESS A ON P.ID = A.PERSON ID;

ID	I	FIRST_NAME	I	LAST_NAME	I	AGE		HOUSE_NO	1	STREET_NAME	1	CITY		STATE	++ PERSON_ID +
		AMAN												KAR	
2		MANOHAR	I	VERMA	I	24		228/C		RRN		BLR		KAR	2
3	I	SWETHA	I	SHARMA	I	21	l	356/D		KKC		BLR		KAR	3
4	I	VIJAY	I	VAISHNAV	I	23		521/F		RJN		BLR		KAR	4
NULL	I	NULL	I	NULL	I	NULL	l	224/I		BTM		BLR		KAR	NULL

FULL OUTER JOIN

• Unfortunately we don't have full outer join in mysql database but this feature is available on oracle database.

SELECT * FROM PERSON P FULL OUTER JOIN ADDRESS A ON P.ID =
A.PERSON_ID;

ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'FULL OUTER JOIN ADDRESS A ON P.ID = A.PERSON_ID' at line 1

ONE-TO-MANY MAPPING

DROP TABLE IF EXISTS PERSON, ADDRESS;

CREATE TABLE PERSON (ID INTEGER PRIMARY KEY, FIRST_NAME VARCHAR(90), LAST_NAME VARCHAR(90), AGE INTEGER);

DROP IF EXISTS TABLE MAIL_ACCOUNT;

CREATE TABLE EMAIL_ID(USERNAME VARCHAR(90), PASSWORD VARCHAR(90), VENDOR VARCHAR(90), PERSON_ID INTEGER, CONSTRAINT MA_FK1 FOREIGN KEY(PERSON_ID) REFERENCES PERSON(ID));

INSERT INTO PERSON VALUES(1, 'AMAN', 'GUPTA', 22);

```
INSERT INTO PERSON VALUES(2, 'MANOHAR', 'VERMA', 24);
INSERT INTO PERSON VALUES(3, 'SWETHA', 'SHARMA', 21);
INSERT INTO PERSON VALUES(4, 'VIJAY', 'VAISHNAV', 23);
INSERT INTO PERSON VALUES(5, 'KUMAR', 'SINHA', 23);

INSERT INTO EMAIL_ID VALUES('AMAN', 'PASS', 'GMAIL', 1);
INSERT INTO EMAIL_ID VALUES('AMAN', 'PASS', 'OUTLOOK', 1);
INSERT INTO EMAIL_ID VALUES('AMAN', 'PASS', 'HOTMAIL', 1);
INSERT INTO EMAIL_ID VALUES('MANOHAR', 'PASS', 'YAHOO', 2);
INSERT INTO EMAIL_ID VALUES('MANOHAR', 'PASS', 'GMAIL', 2);
INSERT INTO EMAIL_ID VALUES('SWETHA', 'PASS', 'GMAIL', 3);
INSERT INTO EMAIL_ID VALUES('KUMAR', 'PASS', 'GMAIL', 4);
INSERT INTO EMAIL_ID (USERNAME, PASSWORD, VENDOR)
VALUES('VIJAY', 'PASS', 'GMAIL');
```

SI	SELECT * FROM PERSON;								
+		+		-+-		+		+	
I	ID	I	FIRST_NAME	I	LAST_NAME	I	AGE	I	
+		+		-+-		+-		+	
	1		AMAN	I	GUPTA		22		
	2		MANOHAR		VERMA	I	24		
	3	1	SWETHA		SHARMA		21		
	4		VIJAY		VAISHNAV		23		
	5		KUMAR		SINHA		23		
+		+		-+-		+-		+	

SELECT * FROM EMAIL_ID;

+		-+-		+		+		+
	USERNAME		PASSWORD		VENDOR	PERS	ON_ID	
<u>.</u>				_				
Τ.		т.		•		T		•
	AMAN		PASS		GMAIL		1	
1	AMAN	I	PASS		OUTLOOK	1	1	I
	AMAN		PASS	I	HOTMAIL		1	
1	MANOHAR	I	PASS	I	YAHOO		2	
1	MANOHAR	I	PASS		GMAIL		2	
	SWETHA		PASS	I	GMAIL		3	
I	KUMAR	I	PASS	I	GMAIL		4	
1	VIJAY	I	PASS		GMAIL		NULL	
+.		-+-		+		+		+

```
SELECT * FROM PERSON WHERE FIRST NAME = 'AMAN';
+---+------
| ID | FIRST NAME | LAST NAME | AGE |
+---+
| 1 | AMAN | GUPTA | 22 |
+---+---+---+
SELECT * FROM EMAIL_ID WHERE USERNAME = 'AMAN';
+----+
+----+
AMAN | PASS | GMAIL |
AMAN | PASS | OUTLOOK | 1 |
AMAN PASS HOTMAIL
+-----
SELECT * FROM EMAIL ID WHERE PERSON ID = 3;
+----+
+----+
| SWETHA | PASS | GMAIL |
```

SELECT * FROM EMAIL_ID WHERE PERSON_ID = (SELECT ID FROM
PERSON WHERE FIRST_NAME = 'AMAN');

SELECT * FROM PERSON WHERE ID = (SELECT PERSON_ID FROM
EMAIL_ID WHERE USERNAME = 'AMAN');

ERROR 1242 (21000): Subquery returns more than 1 row

SELECT * FROM PERSON, EMAIL_ID WHERE PERSON.ID =
EMAIL_ID.PERSON_ID;

++	-+	-+	+	-+	-+	-+
ID FIRST_NAME	LAST_NAME	AGE	USERNAME	PASSWORD	VENDOR	PERSON_ID
++	-+	-+	+	-+	-+	-+
1 AMAN	GUPTA	2	22 AMAN	PASS	GMAIL	1
1 AMAN	GUPTA	2	22 AMAN	PASS	OUTLOOK	1
1 AMAN	GUPTA	2	22 AMAN	PASS	HOTMAIL	1
2 MANOHAR	VERMA	2	4 MANOHAR	PASS	YAH00	2
2 MANOHAR	VERMA	2	4 MANOHAR	PASS	GMAIL	2
3 SWETHA	SHARMA	2	1 SWETHA	PASS	GMAIL	3
4 VIJAY	VAISHNAV	2	3 KUMAR	PASS	GMAIL	4
++	-+	-+	+	-+	-+	-+

SELECT * FROM PERSON P, EMAIL_ID M WHERE P.ID =
M.PERSON_ID;

SELECT * FROM PERSON P INNER JOIN EMAIL_ID M ON P.ID =
M.PERSON_ID;

SELECT * FROM PERSON P LEFT OUTER JOIN EMAIL_ID M ON P.ID =
M.PERSON_ID;

+-	+		-+		-+-		۲.		-+		-+		-+	+
I	ID	FIRST_NAME	I	LAST_NAME	I	AGE		USERNAME	I	PASSWORD	I	VENDOR	PE	RSON_ID
+-	+		-+		+-		+ •		-+		+		-+	+
I	1	AMAN	I	GUPTA	I	22		AMAN	I	PASS	I	GMAIL	1	1
I	1	AMAN	I	GUPTA	I	22		AMAN	I	PASS		OUTLOOK		1
1	1	AMAN	I	GUPTA	I	22		AMAN	١	PASS	١	HOTMAIL	1	1
1	2	MANOHAR	I	VERMA	I	24		MANOHAR	I	PASS	١	YAHOO	1	2
I	2	MANOHAR	I	VERMA	I	24		MANOHAR	I	PASS	١	GMAIL	I	2
I	3	SWETHA	I	SHARMA	I	21		SWETHA	I	PASS	I	GMAIL	1	3
I	4	VIJAY	I	VAISHNAV	I	23		KUMAR	I	PASS	I	GMAIL	1	4
I	5	KUMAR	I	SINHA	I	23		NULL	I	NULL	I	NULL	I	NULL
_ _			- +		- 4 -		. .		- +				. 4	

SELECT * FROM PERSON P RIGHT OUTER JOIN EMAIL_ID M ON P.ID = M.PERSON ID;

+	+	-++		+	+	+
ID	LAST_NAME	AGE	USERNAME	PASSWORD	VENDOR	PERSON_ID
+	+	-++		+	+	
1 AMAN	GUPTA	22	AMAN	PASS	GMAIL	1
1 AMAN	GUPTA	22	AMAN	PASS	OUTLOOK	1
1 AMAN	GUPTA	22	AMAN	PASS	HOTMAIL	1
2 MANOHAR	VERMA	24	MANOHAR	PASS	YAHOO	2
2 MANOHAR	VERMA	24	MANOHAR	PASS	GMAIL	2
3 SWETHA	SHARMA	21	SWETHA	PASS	GMAIL	3
4 VIJAY	VAISHNAV	23	KUMAR	PASS	GMAIL	4
NULL NULL	NULL	NULL	VIJAY	PASS	GMAIL	NULL
+	+	-+		+	+	

ONE-TO-ONE

DROP TABLE IF EXISTS PERSON;

CREATE TABLE PERSON (ID INTEGER PRIMARY KEY, FIRST_NAME VARCHAR(90), AGE INTEGER);

• Because of ID column is PRIMARY KEY PERSON table can become a parent to child table.

DROP TABLE IF EXISTS ADDRESS;

CREATE TABLE ADDRESS(HOUSE_NO VARCHAR(90), STREET_NAME VARCHAR(90), CITY VARCHAR(90), STATE VARCHAR(90), PERSON_ID INTEGER UNIQUE, CONSTRAINT ADDRESS_FK1 FOREIGN KEY(PERSON_ID) REFERENCES PERSON(ID));

```
FSD Training Program
```

```
INSERT INTO PERSON VALUES(1, 'AMAN', 'GUPTA', 22);
INSERT INTO PERSON VALUES(2, 'MANOHAR', 'VERMA', 24);
INSERT INTO PERSON VALUES(3, 'SWETHA', 'SHARMA', 21);
INSERT INTO PERSON VALUES(4, 'VIJAY', 'VAISHNAV', 23);
INSERT INTO PERSON VALUES(5, 'KUMAR', 'SINHA', 23);
SELECT * FROM PERSON;
+---+
| ID | FIRST NAME | LAST NAME | AGE |
+---+------
2 | MANOHAR | VERMA | 24 |
  3 | SWETHA | SHARMA | 21 |
  4 | VIJAY | VAISHNAV | 23 |
  5 | KUMAR | SINHA | 23 |
+---+
INSERT INTO ADDRESS VALUES('185/A', 'BSK', 'BLR', 'KAR',
1);
Query OK, 1 row affected (0.01 sec)
INSERT INTO ADDRESS VALUES('224/B', 'JPN', 'BLR', 'KAR',
1);
ERROR 1062 (23000): Duplicate entry '1' for key
'address.PERSON ID'
```

- FOREIGN KEY is a UNIQUE so we can't insert duplicates.
- One record of PERSON mapping to only one record of ADDRESS. So we call it as one to one mapping.

```
INSERT INTO ADDRESS VALUES('185/A', 'BSK', 'BLR', 'KAR',
1);
Query OK, 1 row affected (0.01 sec)
INSERT INTO ADDRESS VALUES('224/B', 'JPN', 'BLR', 'KAR',
1);
ERROR 1062 (23000): Duplicate entry '1' for key
'address.PERSON ID'
INSERT INTO ADDRESS VALUES('228/C', 'RRN', 'BLR', 'KAR',
2);
Query OK, 1 row affected (0.01 sec)
INSERT INTO ADDRESS VALUES('356/D', 'KKC', 'BLR', 'KAR',
3);
Query OK, 1 row affected (0.00 sec)
INSERT INTO ADDRESS VALUES('521/F', 'RJN', 'BLR', 'KAR',
4);
Query OK, 1 row affected (0.01 sec)
INSERT INTO ADDRESS VALUES('652/G', 'KRL', 'BLR', 'KAR',
6);
ERROR 1452 (23000): Cannot add or update a child row: a
foreign key constraint fails (`mysql notes`.`address`,
CONSTRAINT `ADDRESS FK1` FOREIGN KEY (`PERSON ID`)
```

REFERENCES `person` (`ID`))

• No PERSON with ID as 6

```
INSERT INTO ADDRESS(HOUSE_NO, STREET_NAME, CITY, STATE)
VALUES('224/I', 'BTM', 'BLR', 'KAR');
```

• We can insert ADDRESS without choosing PERSON_ID because this column is UNIQUE and allows NULL values.

```
SELECT * FROM PERSON;
SELECT * FROM ADDRESS;
```

SELECT * FROM PERSON;

+	+		+		+-		+
1	D	FIRST_NAME	I	LAST_NAME		AGE	I
+	+		+		+-		+
	1	AMAN	I	GUPTA	I	22	I
I	2	MANOHAR	I	VERMA	I	24	
1	3	SWETHA		SHARMA		21	
	4	VIJAY		VAISHNAV		23	
1	5	KUMAR		SINHA	I	23	
+	+		+		+-		+
5 r	'OWS	in set (0.0	0	sec)			

```
SELECT * FROM ADDRESS;
+----+
| HOUSE NO | STREET NAME | CITY | STATE | PERSON ID |
+----+
2 |
| 356/D | KKC
         3 l
+----+
SELECT * FROM PERSON WHERE FIRST NAME = 'AMAN';
+---+
| ID | FIRST NAME | LAST NAME | AGE |
+---+
| 1 | AMAN | GUPTA | 22 |
SELECT * FROM ADDRESS WHERE HOUSE NO = '185/A';
+----+
| HOUSE_NO | STREET_NAME | CITY | STATE | PERSON_ID |
+----+
```

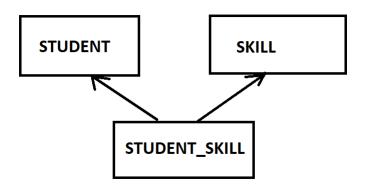
-----+

```
SELECT * FROM ADDRESS WHERE PERSON ID = 3;
+----+
| HOUSE NO | STREET NAME | CITY | STATE | PERSON ID |
+----+
| 356/D | KKC | BLR | KAR |
+----+
SELECT * FROM ADDRESS WHERE PERSON ID = (SELECT ID FROM
PERSON WHERE FIRST_NAME = 'AMAN');
+----+
| HOUSE_NO | STREET_NAME | CITY | STATE | PERSON_ID |
+----+
+----+
SELECT * FROM PERSON WHERE ID = (SELECT PERSON ID FROM
ADDRESS WHERE HOUSE_NO = '185/A');
+---+
| ID | FIRST_NAME | LAST_NAME | AGE |
+---+
+---+
```

MANY-TO-MANY MAPPING

```
DROP TABLE IF EXISTS STUDENT;
CREATE TABLE STUDENT(ID INTEGER UNIQUE, FIRST NAME
VARCHAR(90), LAST NAME VARCHAR(90));
DROP TABLE IF EXISTS SKILL;
CREATE TABLE SKILL(ID INTEGER UNIQUE, NAME VARCHAR(90));
INSERT INTO STUDENT VALUES(1, 'AMAN', 'GUPTA');
INSERT INTO STUDENT VALUES(2, 'MANOHAR', 'VERMA');
INSERT INTO STUDENT VALUES(3, 'JAGAN', 'REDDY');
INSERT INTO STUDENT VALUES(4, 'KUMAR', 'SINHA');
INSERT INTO SKILL VALUES(1, 'C');
INSERT INTO SKILL VALUES(2, 'C++');
INSERT INTO SKILL VALUES(3, 'JAVA');
INSERT INTO SKILL VALUES(4, 'SQL');
DROP TABLE IF EXTISTS STUDENT_SKILL;
CREATE TABLE STUDENT SKILL(STUDENT ID INTEGER, SKILL ID
INTEGER, CONSTRAINT SS FK1 FOREIGN KEY(STUDENT ID)
REFERENCES STUDENT(ID), CONSTRAINT SS FK2 FOREIGN
```

KEY(SKILL ID) REFERENCES SKILL(ID));



```
INSERT INTO STUDENT SKILL VALUES(1, 1);
INSERT INTO STUDENT_SKILL VALUES(1, 2);
INSERT INTO STUDENT_SKILL VALUES(2, 2);
INSERT INTO STUDENT_SKILL VALUES(3, 1);
INSERT INTO STUDENT_SKILL VALUES(3, 4);
SELECT * FROM STUDENT;
+----+
+----+
   1 | AMAN | GUPTA
   2 | MANOHAR | VERMA
   3 | JAGAN | REDDY
   4 | KUMAR | SINHA
 -----+
4 rows in set (0.00 sec)
```

```
FSD Training Program

SELECT * FROM SKILL;

+----+
```

```
| ID | NAME |
+----+
| 1 | C |
| 2 | C++ |
| 3 | JAVA |
| 4 | SQL |
+----+
4 rows in set (0.00 sec)
SELECT * FROM STUDENT_SKILL;
+-----+
| STUDENT_ID | SKILL_ID |
       1 | 1 |
       1 2 |
       2 | 2 |
       3 | 1 |
       3 | 4 |
```

```
SELECT NAME FROM SKILL WHERE ID IN

(SELECT SKILL_ID FROM STUDENT_SKILL WHERE STUDENT_ID =

(SELECT ID FROM STUDENT WHERE FIRST_NAME = 'AMAN'));
```

<pre>SELECT FIRST_NAME FROM STUDENT WHERE ID IN (SELECT STUDENT_ID FROM STUDENT_SKILL WHERE SKILL_ID = (SELECT ID FROM SKILL WHERE NAME = 'C')); ++</pre>
FIRST_NAME
++
AMAN
JAGAN
++
SELECT * FROM STUDENT, STUDENT_SKILL, SKILL WHERE
STUDENT.ID = STUDENT_SKILL.STUDENT_ID AND STUDENT_SKILL.SKILL_ID = SKILL.ID;
SELECT * FROM STUDENT S INNER JOIN STUDENT_SKILL SS
<pre>ON S.ID = SS.STUDENT_ID INNER JOIN SKILL SK ON SS.SKILL_ID = SK.ID;</pre>
To generate database diagram in mysql(ER diagram)
Database -> Reverse Engineer -> choose your connection -> login -> hit continue -> select database schema -> hit continue -> continue again -> make sure items are checked -> execute -> again hit continue -> close
THE END