***DATA BASE ASSIGNMENT***

\* What do you understand By Database

A database is like a digital filing cabinet where data is stored and organized so it can be easily accessed, managed, and updated. It’s used to keep track of information in a structured way.

\* What is Normalization?

Normalization is a way to organize a database to reduce duplicate data and ensure accuracy. It involves splitting data into smaller, related tables.

\* What is Difference between DBMS and RDBMS?

DBMS (Database Management System) stores data as files without relationships between them. It supports single users and has lower security.

RDBMS (Relational Database Management System) stores data in tables with relationships between them. It supports multiple users, offers higher security, and allows for data normalization.

\* What is MF Cod Rule of RDBMS Systems?

Codd’s Rules are 13 guidelines that define what makes a true RDBMS. They ensure data is stored in tables and managed relationally.

\* What do you understand By Data Redundancy?

Data redundancy is when the same data is stored in multiple places. This can waste space and cause inconsistencies.

\* What is DDL Interpreter?

A DDL (Data Definition Language) Interpreter processes commands that define or change the structure of a database, like creating or altering tables.

\* What is DML Compiler in SQL?

A DML (Data Manipulation Language) Compiler converts commands like INSERT, UPDATE, and DELETE into instructions the database can execute.

\* What is SQL Key Constraints writing an Example of SQL Key Constraints

SQL key constraints are rules applied to columns in a database table to enforce data integrity and uniqueness. Common types include PRIMARY KEY, FOREIGN KEY, UNIQUE, and CHECK constraints.

Example of SQL Key Constraints:

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50) ,

LastName VARCHAR(50) ,

Email VARCHAR(100) UNIQUE KEY,

DepartmentID INT,

FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID) );

\* What is save Point? How to create a save Point write a Query?

A **SAVEPOINT** in SQL is a marker within a transaction that allows you to roll back to a specific point without affecting the entire transaction.

BEGIN; SAVEPOINT sp1; ROLLBACK TO sp1; COMMIT;

\* What is trigger and how to create a Trigger in SQL?

A \*\*trigger\*\* in SQL is an automatic action that executes in response to events like inserting, updating, or deleting records in a table. It helps maintain data integrity and enforce business rules.

CREATE TABLE tiggar\_01(

tid int,

tname varchar (30),

tsubject varchar(30),

tcity varchar (30),

tprice int ,

tim\_date timestamp,

action\_perform varchar (80),);

***assignment 1***

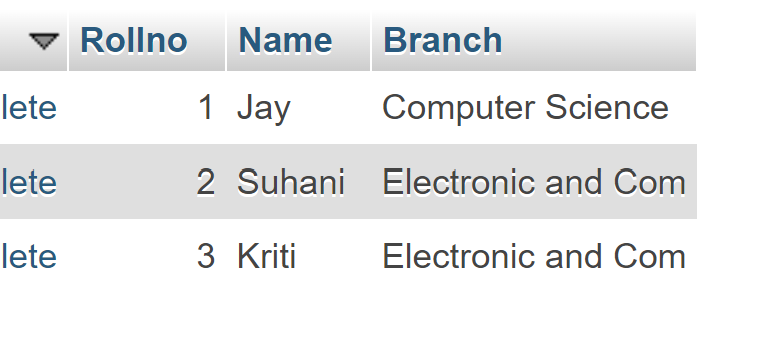
CREATE TABLE student(Rollno int PRIMARY key AUTO\_INCREMENT NOT null, Name varchar (30), Branch varchar (30));

INSERT INTO student (Rollno,Name,Branch)

values (1,"Jay","Computer Science"),

(2,"Suhani","Electronic and Com"),

(3,"Kriti","Electronic and Com");



CREATE TABLE Exam

(Rollno int ,

S\_code varchar (30),

Marks int ,

P\_code varchar (30),

FOREIGN KEY(Rollno) REFERENCES student (Rollno) );

INSERT INTO exam (Rollno,S\_code,Marks,P\_code)

values (1,"CS11",50,"CS"),

(1,"CS12",60,"CS"),

(2,"EC101",66,"EC"),

(2,"EC102",70,"EC"),

(3,"EC101",45,"EC"),

(3,"EC102",50,"EC");

A screenshot of a computer code

Description automatically generated

***assignment 2***

CREATE TABLE info

(FirstName varchar(30),

LastName varchar(30),

Address varchar (90),

City varchar (50),

Age int

);

insert into info (FirstName,LastName,Address,City,Age)

values ("Mickey","Mouse","123 Fantasy Way","Anaheim",73),

("Bat","Man","321 Cavern Ave","Gotham",54),

("Wonder","Woman","987 Truth Way","Paradise",39),

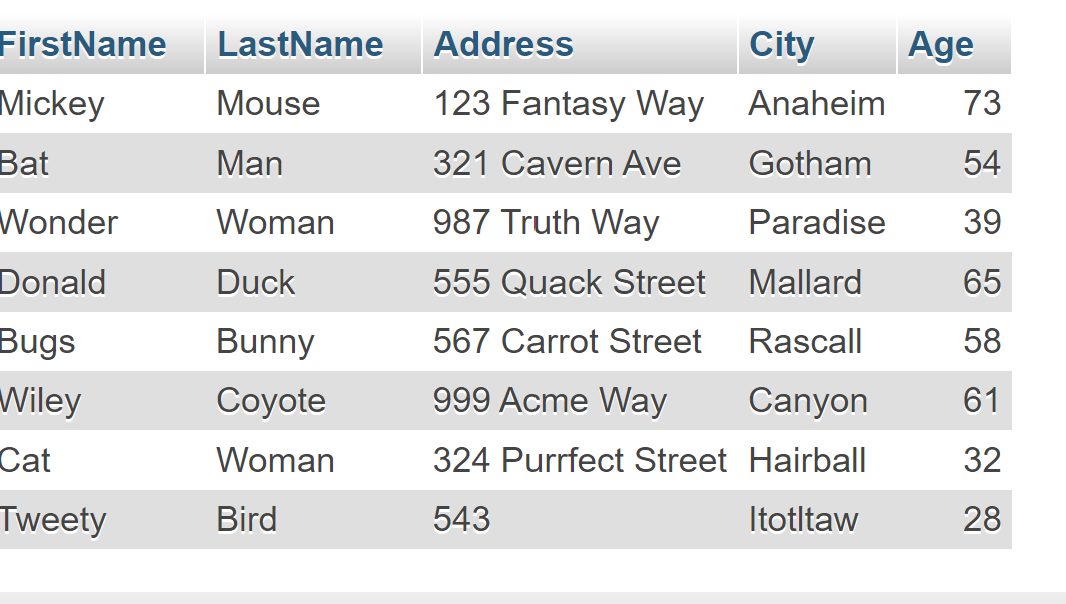
("Donald","Duck","555 Quack Street","Mallard",65),

("Bugs","Bunny","567 Carrot Street","Rascall",58),

("Wiley","Coyote","999 Acme Way","Canyon",61),

("Cat","Woman","324 Purrfect Street","Hairball",32),

("Tweety","Bird","543 ","Itotltaw",28);



***ASSIGNMENT 3***

CREATE TABLE employee

(Employee\_id int PRIMARY KEY AUTO\_INCREMENT,

First\_name varchar (30),

Last\_name varchar (50),

salary bigint,

Joining\_date datetime,

Department varchar (50)

);

insert into employee (Employee\_id,

First\_name,

Last\_name,

salary,

Joining\_date,

Department)

VALUES (1,"John","Abraham",1000000,"2013-01-01 12:00","Banking"),

(2,"Michael","Clarke",800000,"2013-01-01 12:00","Insurance");

(3,"Roy","Thomas",700000,"2013-02-01 12:00","Banking"),

(4,"Tom","Jose",600000,"2013-02-01 12:00","Insurance"),

(5,"Jerry","Pinto",650000,"2013-02-01 12:00","Insurance"),

(6,"Philip","Mathew",750000,"2013-01-01 12:00","Services"),

(7,"Testname1","123",650000,"2013-01-01 12:00","Services"),

(8,"Testname2","Lname%",600000,"2013-02-01 12:00","Insurance");



CREATE table incentive

(Employee\_ref\_id int,

Incentive\_date date,

Incentive\_amount bigint,

FOREIGN key(Employee\_ref\_id) REFERENCES employee (Employee\_id)

);

INSERT INTO incentive (Employee\_ref\_id,

Incentive\_date,

Incentive\_amount)

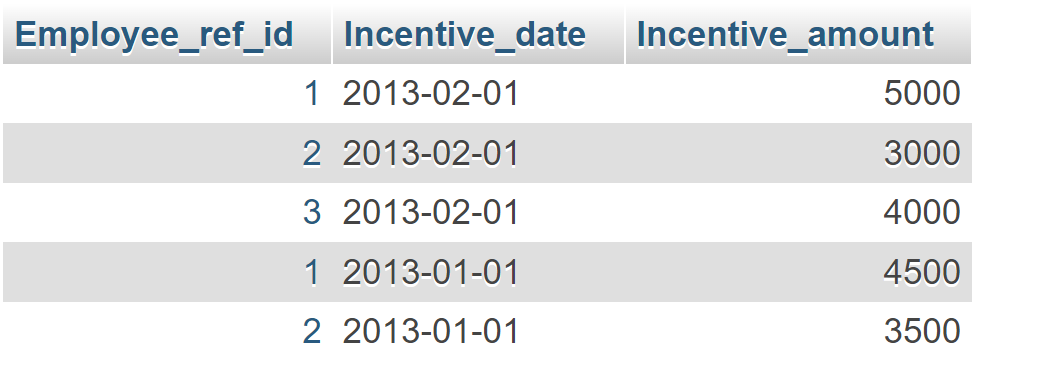
VALUES(1,"2013-02-01",5000),

(2,"2013-02-01",3000),

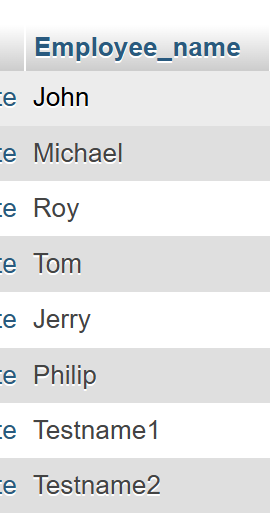
(3,"2013-02-01",4000),

(1,"2013-01-01",4500),

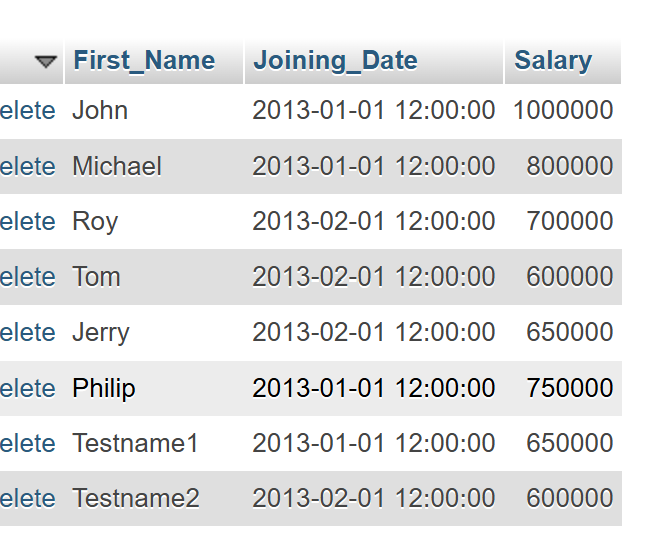
(2,"2013-01-01",3500);



1. a) SELECT First\_name as Employee\_name FROM employee;



1. SELECT First\_Name,Joining\_Date,Salary from employee;



1. SELECT First\_name FROM employee ORDER BY First\_name asc;



SELECT salary FROM employee ORDER BY salary DESC;

A screenshot of a computer

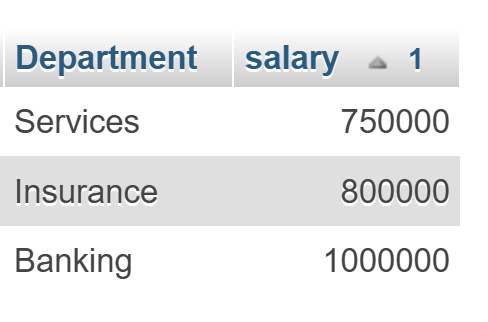
Description automatically generated

1. select First\_name from employee WHERE First\_name like 'J%';

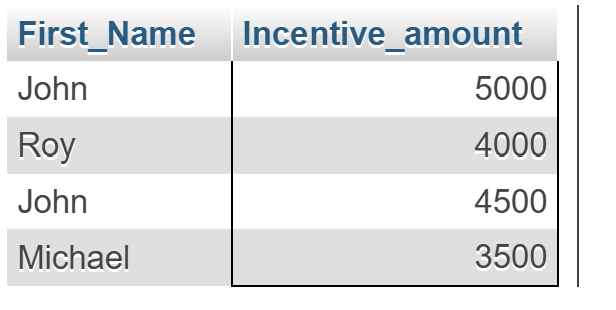
A close-up of a name

Description automatically generated

1. SELECT Department, MAX(salary) AS salary FROM employee GROUP BY Department ORDER BY salary ASC;



1. SELECT First\_name, Incentive\_amount FROM employee JOIN incentive ON Employee\_id = Employee\_ref\_id WHERE Incentive\_amount > 3000;



1. CREATE TABLE trigger\_01

(tEmployee\_id int PRIMARY KEY ,

tFirst\_name varchar (30),

tLast\_name varchar (50),

tsalary bigint,

tJoining\_date datetime,

tDepartment varchar (50),

tim\_date timestamp,

action\_perform (50)

);

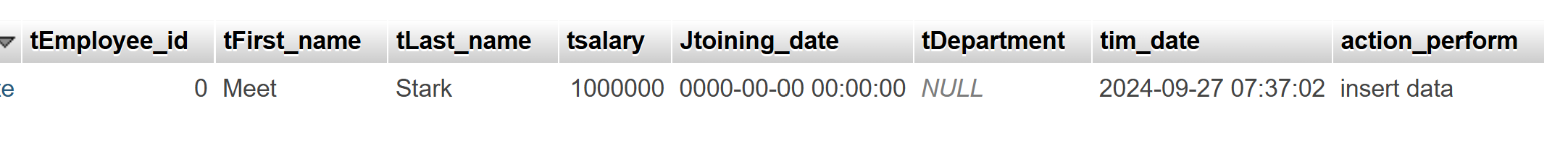
CREATE trigger meet1 BEFORE insert on employee for each row

insert into trigger\_01 (tEmployee\_id,tFirst\_name,tLast\_name,tsalary,tDepartment,Jtoining\_date,action\_perform)

VALUES (new.Employee\_id,new.First\_name,new.Last\_name,new.Salary,new.Joining\_date,new.department,"insert data");

Inserted data🡪 insert into employee(First\_name,Last\_name,salary,Department)

values ("Meet","Stark",1000000,"ITC");



***Assignment 4***

CREATE TABLE salsperson1

(SNo bigint PRIMARY KEY,

SName varchar (30),

City varchar (30),

COMN varchar(30) );

INSERT INTO salsperson1 (SNo,SName,City,COMN)

values (1001,"Peel","London",.12),

(1002,"Serres","San Jose",.13),

(1004,"Motika","London",.11),

(1007,"Rafkin","Barcelona",.15),

(1003,"Axelrod","New York",.1);



CREATE TABLE customer

(CNM bigint PRIMARY KEY,

CName varchar (50),

City varchar (50),

Rating bigint ,

SNo bigint,

FOREIGN KEY(SNo)REFERENCES salsperson2 (SNo));

INSERT INTO customer6(CNM,CName,City,Rating,SNo)

VALUES (201,"Hoffman","London",100,1001),

(202,"Giovanne","Roe",200,1003),

(203,"Liu","San Jose",300,1002),

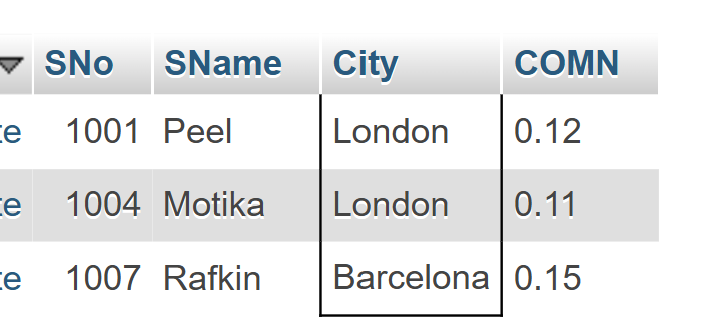
(204,"Grass","Barcelona",100,1002),

(206,"Clemens","London",300,1007),

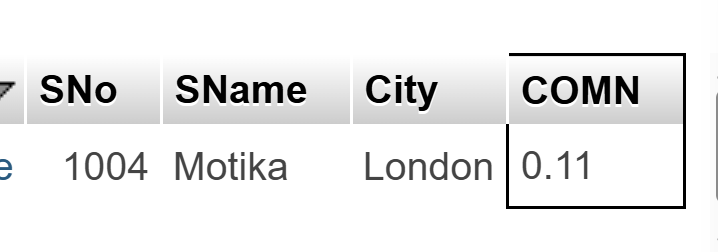
(207,"Pereira","Roe",100,1004);



1. SELECT SName,City from salsperson2 where City = "London" and COMN > 0.12;
2. SELECT \* from salsperson2 where City = "Barcelona" or City = "London";



1. SELECT \* FROM salsperson2 WHERE COMN > 0.10 AND COMN < 0.12;



1. SELECT \* FROM customer6 WHERE (Rating > 100 AND City != "Roe") OR City = "Roe";

