MY SQL PRACTICAL

- ***STUDENT NAME :-JAY RANA***
- ***ENROLLMENT NO. :-23BCT0013***
- ***SUBJECT :-DATABASE DESIGN USING (MYSQL)***

1.Insert a new student with the following details: ID = 41, Name = 'Jane Smith', Age =22, and Grade = 'A' into the LEARNERS_ table.

INPUT:

```
create database college;
use college;

create table student(
std_id int primary key not null,
std_name varchar(50),
std_age int,
grade varchar(1)
);
insert into student(std_id,std_name,std_age,grade)
values(2,'jane smith',22,'A');
select * from student;
```

OUTPUT:

std_id	std_name	std_age	grade
2	jane smith	22	A
NULL	NULL	NULL	NULL

2. **Insert a record into an employee_S table with columns emp_id, emp_name,department, and salary. Insert emp_id = 261, emp_name**

= 'Alice', department = 'HR', and

salary = 50000.

INPUT:

```
create database company;
use company;

create table employees(
emp_id int primary key,
emp_name varchar(50),
department varchar(10),
salary int
);

insert into employees(emp_id,emp_name,department,salary)
values(101,'Alice','HR',50000);
select * from employees;
```

OUTPUT:

emp_id	emp_name	departme...	salary
101	Alice	HR	50000
NULL	NULL	NULL	NULL

3. Insert multiple records into the student table in a single query. Insert one student with ID = 42, Name = 'Mark', Age = 23, Grade = 'B' and another student with ID = 43, Name =

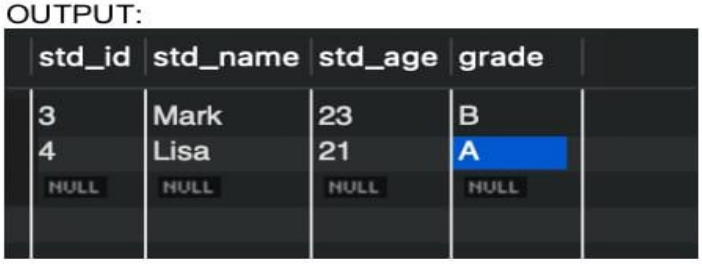
'Lisa', Age = 21, Grade = 'A'.

INPUT:

```
create database college;
use college;

create table student(
std_id int primary key not null,
std_name varchar(50),
std_age int,
grade varchar(1)
);
insert into student(std_id,std_name,std_age,grade)
values(3,'Mark',23,'B'),
(4,'Lisa',21,'A');
select * from student;
```

OUTPUT:



std_id	std_name	std_age	grade
3	Mark	23	B
4	Lisa	21	A
NULL	NULL	NULL	NULL

4. Add a new book to the books table with BookID = 101, Title = 'MySQL Essentials',Author = 'John Doe', Price = 29.99.

INPUT:

```
CREATE DATABASE store;
USE store;
CREATE TABLE books(
bookID INT PRIMARY KEY,
Title VARCHAR(50),
Author VARCHAR(50),
Price decimal(10,2)
);
INSERT INTO books (BookID, Title, Author, Price)
VALUES (101, 'MySQL Essentials', 'John Doe', 29.99);
SELECT * FROM books;
```

OUTPUT:

bookID	Title	Author	Price
101	MySQL Essentials	John Doe	29.99
NULL	NULL	NULL	NULL

5. Retrieve all the students from the LEARNERS_ table.

```
create database college;
use college;

create table student(
std_id int primary key not null,
std_name varchar(50),
std_age int,
grade varchar(1)
);
insert into student(std_id,std_name,std_age,grade)
values(3,'Mark',23,'B'),
(4,'Lisa',21,'A');
select * from student;
```

OUTPUT:

std_id	std_name	std_age	grade
3	Mark	23	B
4	Lisa	21	A
NULL	NULL	NULL	NULL

6. Get the names and majors of all students who are 21 years old or older from the student table.

INPUT:

```
create database college;
use college;
CREATE TABLE student (
    ID INT PRIMARY KEY,
    std_Name VARCHAR(50),
    Age INT,
    Major VARCHAR(50)
);
INSERT INTO student (ID,std_Name, Age, Major)
VALUES
(1, 'bhumi', 22, 'Computer Science'),
(2, 'jhanvi', 20, 'Mathematics'),
(3, 'suchi', 23, 'Physics'),
(4, 'Lisa', 21, 'Engineering'),
(5, 'Mitro', 19, 'Biology');

SELECT std_Name,Major
FROM student
WHERE Age >= 21;
```

OUTPUT:

std_Name	Major
bhumi	Computer Science
suchi	Physics
Lisa	Engineering

7. Select all books from the books table that cost more than 20 dollars.

INPUT:

```
CREATE DATABASE store;
USE store;
CREATE TABLE books(
bookID INT PRIMARY KEY,
Title VARCHAR(50),
Author VARCHAR(50),
Price decimal(10,2)
);
INSERT INTO books (BookID, Title, Author, Price)
VALUES (101, 'MySQL Essentials', 'John Doe', 29.99),
(102,'The white tiger','Arvind Adiga',19.23);

SELECT * FROM books
WHERE Price > 20;
```

OUTPUT:

bookID	Title	Author	Price
101	MySQL Essentials	John Doe	29.99
NULL	NULL	NULL	NULL

8. **Retrieve all customers who have placed at least one order from the orders table. Display their CustomerID, CustomerName, and the TotalAmount of their first order (earliest by date).**

```sql
CREATE TABLE customers (
    CustomerID INT PRIMARY KEY,
    CustomerName VARCHAR(100)
);
INSERT INTO customers (CustomerID, CustomerName)
VALUES
(1, 'jhanvi'),
(2, 'bhumi'),
(3, 'Charmi');
```

```sql
CREATE TABLE orderss (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2),
    FOREIGN KEY (CustomerID) REFERENCES customers(CustomerID)
);
INSERT INTO orderss (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES
(1, 1, '2024-01-15', 150.50),
(2, 1, '2024-01-05', 300.75),
(3, 2, '2024-01-20', 100.00);
```

OUTPUT:

CustomerID	CustomerName	TotalAmount
1	jhanvi	300.75
2	bhumi	100.00

9. List the names of employees who earn more than the average salary in the employees table.

```sql
CREATE DATABASE xyz_companay;
use xyz_companay;
CREATE TABLE employeees (
    EmpID INT PRIMARY KEY,
    emp_Name VARCHAR(50),
    Salary INT
);
INSERT INTO employeees (EmpID,emp_Name, Salary)
VALUES
(1, 'Jhanvi', 50000),
(2, 'bhumi', 60000),
(3, 'jay', 50000),
(4, 'Emily', 7000),
(5, 'akash', 47000);

SELECT emp_Name
FROM employeees
WHERE Salary > (SELECT AVG(Salary) FROM employeees);
```

OUTPUT:

emp_Name
Jhanvi
bhumi
jay
akash

10. **Retrieve all products from the products table that have never been ordered. Display the ProductID, ProductName, and Price.**

```sql
CREATE DATABASE flifcart;
USE flifcart;
CREATE TABLE products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Price DECIMAL(10, 2)
);
INSERT INTO products (ProductID, ProductName, Price)
VALUES
(1, 'jeans', 799.99),
(2, 'kurta', 899.99),
(3, 'shoes', 499.99);
CREATE TABLE orders(
    OrderID INT primary key,
    ProductID INT,
    Quantity INT,
    FOREIGN KEY (ProductID) REFERENCES products(ProductID)
);
INSERT INTO orders (OrderID, ProductID, Quantity)
VALUES
(1, 1, 2),
(2, 2, 1);

SELECT p.ProductID, p.ProductName, p.Price
FROM products p
LEFT JOIN orders o ON p.ProductID = o.ProductID
WHERE o.ProductID IS NULL;
```

OUTPUT:

ProductID	ProductName	Price
3	shoes	499.99

11. **For each CustomerID, show the total number of orders they have placed, the total amount spent, and the average amount spent per order.**

```
CREATE DATABASE amazon;
USE amazon;
CREATE TABLE orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2)
);
INSERT INTO orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES
(1, 1, '2024-01-10', 150.50),
(2, 1, '2024-02-15', 200.75),
(3, 2, '2024-01-20', 600.25),
(4, 3, '2024-02-25', 400.00),
(5, 1, '2024-03-01', 100.00);

SELECT CustomerID,
       COUNT(OrderID) AS TotalOrders,
       SUM(TotalAmount) AS TotalAmountSpent,
       AVG(TotalAmount) AS AvgAmountPerOrder
FROM orders
GROUP BY CustomerID;
```

OUTPUT:

CustomerID	TotalOrders	TotalAmountSp...	AvgAmountPerOrd...
1	3	451.25	150.416667
2	1	600.25	600.250000
3	1	400.00	400.000000

12. Update the age of the student whose ID = 41 to 21 in the LEARNERS_ table.

INPUT:

```
create database college;
use college;
CREATE TABLE students (
    std_ID INT PRIMARY KEY,
    std_Name VARCHAR(50),
    Age INT
);
INSERT INTO students (std_ID,std_Name,Age)
VALUES
(1, 'bhumi', 20),
(2, 'suchi', 22),
(3, 'jay', 23);

UPDATE students
SET Age = 21
WHERE std_ID = 1;
SELECT * FROM students;
```

OUTPUT:

std_ID	std_Name	Age
1	bhumi	21
2	suchi	22
3	jay	23
NULL	NULL	NULL

13. Increase the price of all books by 10% in the books table.

INPUT:

```
CREATE DATABASE store;
USE store;
CREATE TABLE books(
bookID INT PRIMARY KEY,
Title VARCHAR(50),
Author VARCHAR(50),
Price decimal(10,2)
);
INSERT INTO books (BookID, Title, Author, Price)
VALUES (101, 'MySQL Essentials', 'John Doe', 29.99),
(102,'The white tiger','Arvind Adiga',19.23);
drop table books;

UPDATE books
SET Price = Price * 1.10;
select * from books;
```

OUTPUT:

bookID	Title	Author	Price
101	MySQL Essentials	John Doe	36.29
102	The white tiger	Arvind Adiga	23.27
NULL	NULL	NULL	NULL

14. Increase the salary of employees who have been in the employees table for more than 5 years by 10%. Assume there's a column HireDate in the table, and only those hired before 2019-09-30 should get the raise.

INPUT:

```
CREATE DATABASE company;
USE company;
CREATE TABLE employees (
    EmployeeID INT PRIMARY KEY,
    Name VARCHAR(50),
    Salary INT,
    HireDate DATE
);
INSERT INTO employees (EmployeeID, Name, Salary, HireDate)
VALUES
(1, 'John', 50000, '2015-08-15'),
(2, 'Jane', 60000, '2020-05-12'),
(3, 'Mark', 40000, '2018-06-23'),
(4, 'Emily', 70000, '2017-11-30'),
(5, 'Tom', 45000, '2019-10-10');

UPDATE employees
SET Salary = Salary * 1.10
WHERE EmployeeID=1;
SELECT * FROM employees;
```

OUTPUT:

EmployeeID	Name	Salary	HireDate
1	John	55000	2015-08-15
2	Jane	60000	2020-05-12
3	Mark	40000	2018-06-23
4	Emily	70000	2017-11-30
5	Tom	45000	2019-10-10
NULL	NULL	NULL	NULL

15. Change the OrderStatus to 'Completed' for all orders in the orders table that were placed before 2023-12-31 and whose TotalAmount is greater than 100.

INPUT:

```
CREATE DATABASE cafe;
USE cafe;
CREATE TABLE orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2),
    OrderStatus VARCHAR(20)
);
INSERT INTO orders (OrderID, CustomerID, OrderDate, TotalAmount, OrderStatus)
VALUES
(1, 1, '2023-11-10', 150.50, 'Pending'),
(2, 2, '2024-01-15', 200.00, 'Pending'),
(3, 3, '2023-12-25', 95.00, 'Pending'),
(4, 4, '2023-10-05', 120.00, 'Pending'),
(5, 5, '2023-08-20', 500.00, 'Pending');
UPDATE orders
SET OrderStatus = 'Completed'
WHERE OrderDate < '2023-12-31'
AND TotalAmount > 100;
SELECT * FROM orders;
```

OUTPUT:

OrderID	CustomerID	OrderDate	TotalAmount	OrderStatus
1	1	2023-11-10	150.50	Completed
2	2	2024-01-15	200.00	Pending
3	3	2023-12-25	95.00	Pending
4	4	2023-10-05	120.00	Completed
5	5	2023-08-20	500.00	Completed
NULL	NULL	NULL	NULL	NULL

16. Delete the student with ID = 41 from the student table.

```
create database college;
use college;

create table student(
std_id int primary key not null,
std_name varchar(50),
std_age int,
grade varchar(1)
);
insert into student(std_id,std_name,std_age,grade)
values(3,'Mark',23,'B'),
(4,'Lisa',21,'A');
select * from student;

DELETE FROM student
WHERE std_id = 3;
```

OUTPUT:

std_id	std_name	std_age	grade
4	Lisa	21	A
NULL	NULL	NULL	NULL

17. Remove all books from the books table that are priced below 30 dollars.

INPUT:

```
CREATE DATABASE Library;
USE Library;
CREATE TABLE books (
    BookID INT PRIMARY KEY,
    Title VARCHAR(100),
    Author VARCHAR(100),
    Price DECIMAL(10, 2)
);
INSERT INTO books (BookID, Title, Author, Price)
VALUES
(101, 'MySQL Essentials', 'John Doe', 29.99),
(102, 'The inheritance of loss', 'Kiran Desai', 12.50),
(103, 'Autobiography of an unknown india', 'Niradh C. Chaudhari', 19.99),
(104, 'The god of small things', 'Arundhati Roy', 10.00);

DELETE FROM books
WHERE Price < 15;
SELECT * FROM books;
```

OUTPUT:

BookID	Title	Author	Price
101	MySQL Essentials	John Doe	29.99
103	Autobiography of an unknown india	Niradh C. Chaudhari	19.99
NULL	NULL	NULL	NULL

18. **Delete all customers who have not placed any orders in the orders table. Ensure no orders exist for those customers before deleting.**

INPUT:

```
CREATE DATABASE cafee;
USE cafee;
CREATE TABLE customers (
    CustomerID INT PRIMARY KEY,
    CustomerName VARCHAR(100)
);
INSERT INTO customers (CustomerID, CustomerName)
VALUES
(1, 'bhumi'),
(2, 'jhanvi'),
(3, 'anamika'),
(4, 'shubham');

CREATE TABLE orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2),
    FOREIGN KEY (CustomerID) REFERENCES customers(CustomerID)
);
INSERT INTO orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES
(1, 1, '2023-01-10', 200.00),
(2, 2, '2023-02-20', 150.00);

DELETE FROM customers
WHERE CustomerID NOT IN (
    SELECT DISTINCT CustomerID FROM orders
);
SELECT * FROM customers;
```

OUTPUT:

CustomerID	CustomerName
1	bhumi
2	jhanvi
NULL	NULL

19. Remove all products from the products table where the StockQuantity is less than 5 and the product has not been ordered in the past 6 months.

INPUT:

```
CREATE DATABASE amazon;
USE amazon;
CREATE TABLE products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    StockQuantity INT,
    Price DECIMAL(10, 2)
);
INSERT INTO products (ProductID, ProductName, StockQuantity, Price)
VALUES
(1, 'Product A', 3, 19.99),
(2, 'Product B', 10, 29.99),
(3, 'Product C', 4, 39.99),
(4, 'Product D', 8, 49.99);

CREATE TABLE orders (
    OrderID INT PRIMARY KEY,
    ProductID INT,
    OrderDate DATE,
    FOREIGN KEY (ProductID) REFERENCES products(ProductID)
);
INSERT INTO orders (OrderID, ProductID, OrderDate)
VALUES
(1, 1, '2023-06-15'),
(2, 2, '2023-10-20'),
(3, 3, '2023-12-10');

SELECT p.*
FROM products p
WHERE p.StockQuantity < 5
AND p.ProductID NOT IN (
    SELECT o.ProductID
    FROM orders o
    WHERE o.OrderDate >= NOW() - INTERVAL 6 MONTH
);
```

OUTPUT:

ProductID	ProductName	StockQuanti...	Price
1	Product A	3	19.99
3	Product C	4	39.99
NULL	NULL	NULL	NULL

20. Create a new table called course_s with the following columns: CourseID, CourseName, and Credits.

INPUT:

```
CREATE DATABASE dept;
USE dept;
CREATE TABLE courses (
    CourseID INT PRIMARY KEY,
    CourseName VARCHAR(50),
    Credits INT
);
SELECT * FROM courses;
```

21. Insert two new courses into the courses table. One with CourseID = 401, CourseName = 'Database Management', and Credits = 3, and another with CourseID

= 402, CourseName = 'Web Development', and Credits = 4.

INPUT:

```
CREATE DATABASE school;
USE school;
CREATE TABLE courses (
    CourseID INT PRIMARY KEY,
    CourseName VARCHAR(50) NOT NULL,
    Credits INT NOT NULL
);

INSERT INTO courses (CourseID, CourseName, Credits)
VALUES
(201, 'Database Management', 3),
(202, 'Web Development', 4);

SELECT * FROM courses;
```

OUTPUT:

CourseID	CourseName	Credits
201	Database Management	3
202	Web Development	4
NULL	NULL	NULL

22. Insert a new record into the orders table with OrderID = 3001, CustomerID = 105,OrderDate = '2024-01-15', and TotalAmount = 250.50. Ensure that the CustomerID exists

in the customers table.

INPUT:

```
USE amazon;
CREATE TABLE orders(
orderid INT PRIMARY KEY,
Customerid INT,
orderdate DATE,
totalamount DECIMAL(10,2)
);
INSERT INTO orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES (3001, 105, '2024-01-15', 250.50);
CREATE TABLE customers (
    CustomerID INT PRIMARY KEY,
    CustomerName VARCHAR(100)
);
INSERT INTO customers (CustomerID, CustomerName)
VALUES
(101, 'bhumi'),
(102, 'jhanvi'),
(105, 'anamika'),
(104, 'shubham');

SELECT * FROM customers
WHERE CustomerID = 105;
```

OUTPUT:

CustomerID	CustomerName
105	anamika
NULL	NULL

23. Find the employee who earns the highest salary in each department. Assume there's a DepartmentID in the employees table.

```sql
CREATE DATABASE yzcompany;
USE yzcompany;
CREATE TABLE employees (
    EmployeeID INT PRIMARY KEY,
    EmployeeName VARCHAR(100),
    DepartmentID INT,
    Salary INT
);
INSERT INTO employees (EmployeeID, EmployeeName, DepartmentID, Salary)
VALUES
(1, 'Alice', 10, 70000),
(2, 'Bob', 10, 80000),
(3, 'Charlie', 20, 90000),
(4, 'David', 20, 85000),
(5, 'Eva', 30, 75000);

SELECT e.EmployeeID, e.EmployeeName, e.DepartmentID, e.Salary
FROM employees e
JOIN (
    SELECT DepartmentID, MAX(Salary) AS MaxSalary
    FROM employees
    GROUP BY DepartmentID
) AS maxSalaries ON e.DepartmentID = maxSalaries.DepartmentID AND e.Salary = maxSalaries.MaxSalary;
```

OUTPUT:

EmployeeID	EmployeeName	DepartmentID	Salary
2	Bob	10	80000
3	Charlie	20	90000
5	Eva	30	75000

24. Display the CustomerID and TotalAmount for the top 5 largest orders placed in the orders table.

INPUT:

```
CREATE DATABASE amazon;
USE amazon;
CREATE TABLE orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2)
);
INSERT INTO orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES
(1, 101, '2024-01-10', 150.75),
(2, 102, '2024-01-11', 200.50),
(3, 103, '2024-01-12', 300.00),
(4, 104, '2024-01-13', 450.00),
(5, 105, '2024-01-14', 500.25),
(6, 106, '2024-01-15', 120.99);

SELECT CustomerID, TotalAmount
FROM orders
ORDER BY TotalAmount DESC
LIMIT 5;
```

OUTPUT:

CustomerID	TotalAmount
105	500.25
104	450.00
103	300.00
102	200.50
101	150.75