SQL Assignment 2

# 1. For an online purchasing database, create entity relationship diagrams. Create a database object from your entity diagram.

Entity Relationship Diagram (ERD) for an online purchasing system might include the following entities:  
 - Customers: Stores customer details like customer ID, name, contact information.  
 - Products: Stores product details such as product ID, name, category, price.  
 - Orders: Stores information about customer orders like order ID, customer ID, order date.  
 - Order\_Items: Stores the products in an order, including quantity and price at the time of purchase.  
 - Payments: Stores payment details such as payment ID, order ID, payment date, amount.  
  
 Example ERD:  
 +--------------+ +------------+ +-------------+  
 | Customers |-------| Orders |--------| Order\_Items |  
 +--------------+ +------------+ +-------------+  
 | | |  
 +-------------+ +--------------+ +------------+  
 | Payments | | Products | | Payments |  
 +-------------+ +--------------+ +------------+

# 2. Create a SQL stored procedure to register the use of the database, complete it with proper validation and transaction rollback and commit.

SQL Stored Procedure to register database usage with validation and transaction handling:  
  
 DELIMITER $$  
 CREATE PROCEDURE RegisterUser(IN p\_username VARCHAR(100), IN p\_password VARCHAR(100))  
 BEGIN  
 DECLARE exit handler for sqlexception  
 BEGIN  
 -- Rollback the transaction in case of an error  
 ROLLBACK;  
 END;  
   
 -- Start a new transaction  
 START TRANSACTION;  
  
 -- Validation check: Ensure the username is not empty  
 IF p\_username IS NULL OR p\_username = '' THEN  
 SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'Username cannot be empty';  
 END IF;  
  
 -- Insert new user record  
 INSERT INTO Users(username, password) VALUES(p\_username, p\_password);  
  
 -- Commit the transaction  
 COMMIT;  
 END $$  
 DELIMITER ;

# 3. List the SQL aggregate functions and demonstrate how to utilize it.

SQL Aggregate Functions:  
 1. COUNT(): Counts the number of rows.  
 2. SUM(): Adds the values in a numeric column.  
 3. AVG(): Computes the average of numeric values.  
 4. MIN(): Returns the minimum value in a column.  
 5. MAX(): Returns the maximum value in a column.  
  
 Example usage:  
 SELECT COUNT(\*) AS TotalOrders FROM Orders;  
 SELECT AVG(price) AS AveragePrice FROM Products;  
 SELECT MAX(price) AS HighestPrice FROM Products;

# 4. Create a pivot query in SQL.

Pivot query in SQL: Pivoting allows us to transform rows into columns.  
 Example: Pivoting the Sales data to show products as columns.  
   
 SELECT ProductID,   
 SUM(CASE WHEN MONTH(OrderDate) = 1 THEN Amount ELSE 0 END) AS January,   
 SUM(CASE WHEN MONTH(OrderDate) = 2 THEN Amount ELSE 0 END) AS February  
 FROM Sales  
 GROUP BY ProductID;

# 5. Describe how to join in SQL with an example.

SQL JOINs combine rows from two or more tables based on a related column.  
 Types of JOINs:  
 1. INNER JOIN: Returns rows when there is a match in both tables.  
 2. LEFT JOIN: Returns all rows from the left table, with matching rows from the right table.  
 3. RIGHT JOIN: Returns all rows from the right table, with matching rows from the left table.  
 4. FULL JOIN: Returns all rows when there is a match in either table.  
  
 Example:   
 SELECT Orders.OrderID, Customers.CustomerName  
 FROM Orders  
 INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

# 6. How to locate the 4th highest value in a column in a row. Create your table.

Example to find the 4th highest value in a column:  
   
 CREATE TABLE Employee (  
 EmployeeID INT PRIMARY KEY,  
 Name VARCHAR(100),  
 Salary DECIMAL(10,2)  
 );  
   
 INSERT INTO Employee (EmployeeID, Name, Salary) VALUES  
 (1, 'Alice', 60000),  
 (2, 'Bob', 50000),  
 (3, 'Charlie', 70000),  
 (4, 'David', 55000),  
 (5, 'Eve', 45000);  
   
 -- Query to find the 4th highest salary:  
 SELECT MAX(Salary) AS FourthHighestSalary  
 FROM Employee  
 WHERE Salary NOT IN (  
 SELECT Salary  
 FROM Employee  
 ORDER BY Salary DESC  
 LIMIT 3  
 );