-- Q1 A Yes

SELECT employeeNumber, firstName, lastName FROM employees WHERE jobTitle = "Sales Rep" AND reportsTo = 1102;

-- Q1 B Yes

SELECT DISTINCT productLine FROM products WHERE productLine LIKE "%Cars";

-- Q2 Yes

SELECT customerNumber, customerName, CASE WHEN country IN ('USA', 'Canada') THEN 'North America' WHEN country IN ('UK', 'France', 'Germany') THEN 'Europe' ELSE 'Other' END AS CustomerSegment FROM Customers;

-- Q3A Yes

SELECT productCode, SUM(quantityOrdered) AS total\_ordered FROM OrderDetails GROUP BY productCode ORDER BY total\_ordered DESC LIMIT 10;

-- Q3B Yes

SELECT MONTHNAME(paymentDate) AS payment\_month, COUNT(\*) AS num\_payments FROM Payments GROUP BY MONTHNAME(paymentDate) HAVING COUNT(\*) > 20 ORDER BY num\_payments DESC;

-- Q4A Yes

USE Customers\_Orders;

DROP TABLE IF EXISTS Customers;

CREATE TABLE Customers (customer\_id INT AUTO\_INCREMENT PRIMARY KEY, first\_name VARCHAR(50) NOT NULL, last\_name VARCHAR(50) NOT NULL, email VARCHAR(255) UNIQUE, phone\_number VARCHAR(20));

-- Q4B Yes

USE Customers\_Orders;

CREATE TABLE Orders (order\_id INT AUTO\_INCREMENT PRIMARY KEY, customer\_id INT, order\_date DATE, total\_amount DECIMAL(10, 2), CONSTRAINT fk\_customer FOREIGN KEY (customer\_id) REFERENCES Customers(customer\_id), CONSTRAINT chk\_total\_amount CHECK (total\_amount > 0));

-- Q5 Yes

DESCRIBE Orders;

SELECT c.country, COUNT(\*) AS order\_count FROM Customers c JOIN Orders o ON c.customerNumber = o.customerNumber GROUP BY c.country ORDER BY order\_count DESC LIMIT 5;

-- Q6 Yes

CREATE TABLE project (EmployeeID INT PRIMARY KEY AUTO\_INCREMENT, FullName VARCHAR(50) NOT NULL, Gender ENUM('Male', 'Female') NOT NULL, ManagerID INT);

INSERT INTO project (EmployeeID, FullName, Gender, ManagerID) VALUES (1, 'Pranaya', 'Male', 3), (2, 'Priyanka', 'Female', 1), (3, 'Preety', 'Female', NULL), (4, 'Anurag', 'Male', 1),(5, 'Sambit', 'Male', 1), (6, 'Rajesh', 'Male', 3), (7, 'Hina', 'Female', 3);

SELECT m.FullName AS `Manager Name`, e.FullName AS `Emp Name` FROM project e JOIN project m ON e.ManagerID = m.EmployeeID;

-- Q7 Yes

DROP TABLE IF EXISTS facility;

CREATE TABLE facility (`Facility\_ID` INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, `Name` VARCHAR(100), `City` VARCHAR(100) NOT NULL, `State` VARCHAR(100), `Country` VARCHAR(100));

SHOW COLUMNS FROM facility;

-- Q8 Yes

CREATE OR REPLACE VIEW product\_category\_sales AS SELECT pl.productLine AS productline, ROUND(SUM(od.quantityOrdered \* od.priceEach), 2) AS total\_sales, COUNT(DISTINCT o.orderNumber) AS number\_of\_orders FROM productlines pl JOIN products p ON pl.productLine = p.productLine JOIN orderdetails od ON p.productCode = od.productCode JOIN orders o ON od.orderNumber = o.orderNumber GROUP BY pl.productLine ORDER BY total\_sales DESC;

SELECT productline, total\_sales, number\_of\_orders FROM product\_category\_sales;

-- Q9 Yes

DELIMITER //

CREATE PROCEDURE Get\_country\_payments (IN input\_year INT, IN input\_country VARCHAR(100)) BEGIN SELECT input\_year AS `Year`, input\_country AS `Country`, CONCAT(ROUND(SUM(p.amount) / 1000), 'K') AS `Total Amount` FROM customers c JOIN payments p ON c.customerNumber = p.customerNumber WHERE YEAR(p.paymentDate) = input\_year AND c.country = input\_country;

END //

DELIMITER ;

CALL Get\_country\_payments(2003, 'France');

-- Q10 A

SELECT c.customerName, COUNT(o.orderNumber) AS Order\_count, DENSE\_RANK() OVER (ORDER BY COUNT(o.orderNumber) DESC) AS order\_frequency\_rk FROM customers c LEFT JOIN orders o ON c.customerNumber = o.customerNumber GROUP BY c.customerNumber, c.customerName ORDER BY Order\_count DESC;

-- Q10 B

WITH monthly\_data AS (SELECT YEAR(orderDate) AS Year, MONTH(orderDate) AS Month, MONTHNAME(orderDate) AS MonthName, COUNT(\*) AS OrderCount, LAG(COUNT(\*), 12) OVER (ORDER BY YEAR(orderDate), MONTH(orderDate)) AS PrevYearCount FROM orders GROUP BY YEAR(orderDate), MONTH(orderDate), MONTHNAME(orderDate)), yearly\_stats AS (SELECT Year, MIN(Year) OVER () AS min\_year, MAX(Year) OVER () AS max\_year FROM monthly\_data GROUP BY Year) SELECT m.Year, m.MonthName AS Month, m.OrderCount AS `Total Orders`, CASE WHEN m.PrevYearCount IS NOT NULL AND y.Year > y.min\_year THEN CONCAT(ROUND((m.OrderCount - m.PrevYearCount)/m.PrevYearCount\*100), '%') WHEN y.Year = y.min\_year AND EXISTS (SELECT 1 FROM monthly\_data m2 WHERE m2.Year = y.min\_year + 1 AND m2.Month = m.Month) THEN CONCAT(ROUND((m.OrderCount - (SELECT OrderCount FROM monthly\_data WHERE Year = y.min\_year + 1 AND Month = m.Month))/m.OrderCount\*100), '%') ELSE 'MADE'END AS `% YoY Change` FROM monthly\_data m JOIN yearly\_stats y ON m.Year = y.Year ORDER BY m.Year, m.Month;

-- Q11 Yes

SELECT productLine, COUNT(\*) AS Total FROM products WHERE buyPrice > (SELECT AVG(buyPrice) FROM products) GROUP BY productLine;

-- Q12 Yes

CREATE TABLE Emp\_EH (EmpID INT PRIMARY KEY, EmpName VARCHAR(100), EmailAddress VARCHAR(100));

DELIMITER //

CREATE PROCEDURE InsertEmp\_EH(IN p\_EmpID INT, IN p\_EmpName VARCHAR(100), IN p\_EmailAddress VARCHAR(100)) BEGIN DECLARE EXIT HANDLER FOR SQLEXCEPTION BEGIN SELECT 'Error occurred' AS Message;

END;

INSERT INTO Emp\_EH (EmpID, EmpName, EmailAddress) VALUES (p\_EmpID, p\_EmpName, p\_EmailAddress);

SELECT 'Record inserted successfully' AS Message;

END //

DELIMITER ;

CALL InsertEmp\_EH(1, 'Duplicate Record', 'duplicate@example.com');

-- Q13 Yes

CREATE TABLE Emp\_BIT (Name VARCHAR(100), Occupation VARCHAR(100), Working\_date DATE, Working\_hours INT);

DELIMITER //

CREATE TRIGGER trg\_BeforeInsert\_PositiveHours BEFORE INSERT ON Emp\_BIT FOR EACH ROW

BEGIN IF NEW.Working\_hours < 0 THEN SET NEW.Working\_hours = ABS(NEW.Working\_hours);

END IF;

END;

//

DELIMITER ;

INSERT INTO Emp\_BIT VALUES

('Robin', 'Scientist', '2020-10-04', 12),

('Warner', 'Engineer', '2020-10-04', 10),

('Peter', 'Actor', '2020-10-04', 13),

('Marco', 'Doctor', '2020-10-04', 14),

('Brayden', 'Teacher', '2020-10-04', 12),

('Antonio', 'Business', '2020-10-04', 11);

INSERT INTO Emp\_BIT VALUES ('Karan', 'QA Lead', '2025-04-23', -8);

SELECT \* FROM Emp\_BIT;