**QUESTION 1**

CREATE TABLE employees (

emp\_id INT PRIMARY KEY,

emp\_name VARCHAR(50),

emp\_email VARCHAR(100),

emp\_department VARCHAR(50),

emp\_salary DECIMAL(10, 2)

);

INSERT INTO employees (emp\_id, emp\_name, emp\_email, emp\_department, emp\_salary)

VALUES (1, 'John Smith', 'john.smith@company.com', 'Sales', 5000.00),

(2, 'Jane Doe', 'jane.doe@company.com', 'Marketing', 6000.00),

(3, 'Mark Lee', 'mark.lee@company.com', 'IT', 7000.00);

**employees:**

| **emp\_id** | **emp\_name** | **emp\_email** |
| --- | --- | --- |
| 1 | John Smith | [john.smith@company.com](mailto:john.smith@company.com) |
| 2 | Jane Doe | [jane.doe@company.com](mailto:jane.doe@company.com) |
| 3 | Mark Lee | [mark.lee@company.com](mailto:mark.lee@company.com) |
| 4 | Sarah Tan | [sarah.tan@company.com](mailto:sarah.tan@company.com) |
| 5 | David Lim | [david.lim@company.com](mailto:david.lim@company.com) |

CREATE TABLE sales (

sale\_id INT PRIMARY KEY,

sale\_date DATE,

customer\_id INT,

product\_id INT,

quantity INT,

total\_price DECIMAL(10, 2)

);

INSERT INTO sales (sale\_id, sale\_date, customer\_id, product\_id, quantity, total\_price)

VALUES (1, '2023-02-17', 101, 201, 2, 100.00),

(2, '2023-02-18', 102, 202, 3, 150.00),

(3, '2023-02-19', 103, 203, 1, 50.00);

**sales:**

| **sale\_id** | **emp\_id** | **sale\_amount** | **sale\_date** |
| --- | --- | --- | --- |
| 1 | 1 | 100 | 2023-02-10 |
| 2 | 1 | 50 | 2023-02-11 |
| 3 | 2 | 75 | 2023-02-10 |
| 4 | 4 | 200 | 2023-02-12 |

To get a list of all employees who did not make any sales, we can use the following SQL query:

SELECT emp\_id, emp\_name

FROM employees

WHERE emp\_id NOT IN (SELECT emp\_id FROM sales)

The results:

| **emp\_id** | **emp\_name** |
| --- | --- |
| 3 | Mark Lee |
| 5 | David Lim |

**QUESTION 2**

SELECT Country, COUNT(CustomerID) AS NumCustomers

FROM Customers

GROUP BY Country

HAVING COUNT(CustomerID) > 3

ORDER BY NumCustomers DESC;

**QUESTION 3**

CREATE PROCEDURE insert\_or\_update\_employee

@emp\_id INT,

@emp\_name VARCHAR(50),

@emp\_email VARCHAR(100),

@emp\_department VARCHAR(50),

@emp\_salary DECIMAL(10, 2)

AS

BEGIN

MERGE employees AS target

USING (VALUES (@emp\_id, @emp\_name, @emp\_email, @emp\_department, @emp\_salary)) AS source (emp\_id, emp\_name, emp\_email, emp\_department, emp\_salary)

ON target.emp\_id = source.emp\_id

WHEN MATCHED THEN

UPDATE SET emp\_name = source.emp\_name, emp\_email = source.emp\_email, emp\_department = source.emp\_department, emp\_salary = source.emp\_salary

WHEN NOT MATCHED THEN

INSERT (emp\_id, emp\_name, emp\_email, emp\_department, emp\_salary)

VALUES (source.emp\_id, source.emp\_name, source.emp\_email, source.emp\_department, source.emp\_salary);

END;

To use this procedure, you can call it with the input values like this

EXEC insert\_or\_update\_employee 1, 'John Doe', 'johndoe@example.com', 'Marketing', 50000.00;

**QUESTION 4**

CREATE TABLE EmployeeDetails (

EmpId INT PRIMARY KEY,

EmpName VARCHAR(50),

EmpEmail VARCHAR(100),

EmpDepartment VARCHAR(50),

EmpSalary DECIMAL(10, 2)

);

INSERT INTO EmployeeDetails (EmpId, EmpName, EmpEmail, EmpDepartment, EmpSalary)

VALUES (1, 'John Doe', 'johndoe@example.com', 'Marketing', 50000.00),

(2, 'Jane Smith', 'janesmith@example.com', 'Sales', 60000.00),

(3, 'Bob Johnson', 'bobjohnson@example.com', 'HR', 55000.00),

(4, 'John Doe', 'johndoe@example.com', 'Marketing', 50000.00),

(5, 'Jane Smith', 'janesmith@example.com', 'Sales', 60000.00);

-- fetch duplicate records

SELECT EmpName, EmpEmail, EmpDepartment, EmpSalary, COUNT(\*) AS NumDuplicates

FROM EmployeeDetails

GROUP BY EmpName, EmpEmail, EmpDepartment, EmpSalary

HAVING COUNT(\*) > 1;

**QUESTION 5**

CREATE TABLE EmployeeDetails (

EmpId INT PRIMARY KEY,

EmpName VARCHAR(50),

EmpEmail VARCHAR(100),

EmpDepartment VARCHAR(50),

EmpSalary DECIMAL(10, 2)

);

INSERT INTO EmployeeDetails (EmpId, EmpName, EmpEmail, EmpDepartment, EmpSalary)

VALUES (1, 'John Doe', 'johndoe@example.com', 'Marketing', 50000.00),

(2, 'Jane Smith', 'janesmith@example.com', 'Sales', 60000.00),

(3, 'Bob Johnson', 'bobjohnson@example.com', 'HR', 55000.00),

(4, 'Mary Wilson', 'marywilson@example.com', 'Finance', 70000.00),

(5, 'Tom Baker', 'tombaker@example.com', 'Marketing', 60000.00);

-- fetch only odd rows

SELECT \*

FROM EmployeeDetails

WHERE EmpId % 2 = 1;

**QUESTION 6**

CREATE FUNCTION CalculateAge (@dob DATE)

RETURNS INT

AS

BEGIN

DECLARE @age INT;

SET @age = DATEDIFF(day, @dob, GETDATE()) / 365;

RETURN @age;

END;

To use this function, you can call it with a date of birth like this

SELECT dbo.CalculateAge('1980-01-01');