**SQL CASE STUDY**

**Task Title: Perform Basic SQL Operations on Employees Table**

**Task Description**

Create and manipulate an employees table with the following columns and constraints:

* **emp\_id (INT)**: PRIMARY KEY, AUTO\_INCREMENT
* **name (VARCHAR(50))**: NOT NULL
* **email (VARCHAR(100))**: UNIQUE
* **department (VARCHAR(30))**: DEFAULT 'General'
* **age (INT)**: CHECK that age is 18 or above
* **city (VARCHAR(30))**
* **salary (INT)**

Assume a second table bonus with the following structure:

* **emp\_id (INT)**
* **bonus\_amount**
* **(INT)**

### 1. Table Creation

* Create the employees table with the above constraints.
* Create the bonus table with the specified columns.

**2. Data Insertion**

* Insert at least 5 sample records into the employees table.
* Insert at least 3 records into the bonus table.

**3. SQL Operations**

**i. SQL Operations**

* 1. Display all records from the employees table.
  2. Update the salary of the employee named 'Arun' to 35000.

**ii. SQL Constraints**

1. Try to insert a record with age below 18. What happens and why?

**iii. SQL Operators**

a. Select all employees from the 'Sales' department.

1. Select employees whose salary is greater than 30000.

**iv. SQL Joins**

1. Write a LEFT JOIN to list all employees along with their bonus amount (if available).

**v. SQL GROUP BY and HAVING**

a. Show total salary for each department.

1. Show average salary for each city where the average salary is above 30000.

**4. Aggregate Functions & DISTINCT**

1. Count the total number of employees.
2. Count how many unique departments are there.

**5. ORDER BY and LIMIT (TOP)**

1. Display the top 3 highest paid employees.

**6. Sub-Query**

a. List employees who earn more than the average salary of all employees.

**7. UNION**

a. Combine and display all employee names and cities in a single column using UNION.

**8. Views**

a. Create a view to show only 'Sales' department employees.

1. Query the view to display its data.

**9. Stored Procedure**

1. Create a stored procedure named GetHighSalary that returns employees with salary above 30000.
2. Call the stored procedure.

**10. Trigger**

1. Create a trigger that logs the name and action type into a log table whenever an employee is deleted.
2. Test the trigger by deleting one employee record.