**DBMS Assignment(Answers) SQL Script**

**NAMES: TETA HUGUETTE**

**ID NUMBER: 28982**

# 1. Creating Database

CREATE DATABASE company\_db;  
USE company\_db;

# 2. Creating Tables

CREATE TABLE departments (  
 department\_id INT PRIMARY KEY,  
 department\_name VARCHAR(100)  
);  
  
CREATE TABLE employees (  
 employee\_id INT PRIMARY KEY,  
 first\_name VARCHAR(50),  
 last\_name VARCHAR(50),  
 email VARCHAR(100),  
 hire\_date DATE,  
 salary DECIMAL(10,2),  
 department\_id INT,  
 FOREIGN KEY (department\_id) REFERENCES departments(department\_id)  
);  
  
CREATE TABLE projects (  
 project\_id INT PRIMARY KEY,  
 project\_name VARCHAR(100),  
 start\_date DATE,  
 end\_date DATE  
);  
  
CREATE TABLE employee\_projects (  
 employee\_id INT,  
 project\_id INT,  
 assigned\_date DATE,  
 PRIMARY KEY (employee\_id, project\_id),  
 FOREIGN KEY (employee\_id) REFERENCES employees(employee\_id),  
 FOREIGN KEY (project\_id) REFERENCES projects(project\_id)  
);

# 3. Inserting Data

-- Departments  
INSERT INTO departments VALUES (1, 'Human Resources');

INSERT INTO departments VALUES (2, 'Finance');

INSERT INTO departments VALUES (3, 'Information Technology');

INSERT INTO departments VALUES (4, 'Marketing');

INSERT INTO departments VALUES (5, 'Legal');

INSERT INTO departments VALUES (6, 'Operations');

INSERT INTO departments VALUES (7, 'Customer Service');

INSERT INTO departments VALUES (8, 'Sales');

INSERT INTO departments VALUES (9, 'Research and Development');

INSERT INTO departments VALUES (10, 'Procurement');  
  
-- Employees  
INSERT INTO employees (employee \_id, first \_name, last \_name, email, hire \_date, salary, department \_id) VALUES  
(101, 'Alice', 'Johnson', 'alice.johnson@company.com', '2015-03-15', 4500.00, 1),  
(102, 'Bob', 'Smith', 'bob.smith@company.com', '2018-06-23', 5200.00, 3),  
(103, 'Carol', 'Adams', 'carol.adams@company.com', '2012-09-10', 6700.00, 2),  
(104, 'David', 'Lee', 'david.lee@company.com', '2020-01-05', 3800.00, 4),  
(105, 'Eve', 'Martins', 'eve.martins@company.com', '2019-12-11', 4000.00, 3),  
(106, 'Frank', 'Green', 'frank.green@company.com', '2017-07-08', 6000.00, 8),  
(107, 'Grace', 'Brown', 'grace.brown@company.com', '2014-11-02', 4900.00, 5),  
(108, 'Hank', 'Wilson', 'hank.wilson@company.com', '2013-02-17', 3100.00, 6),  
(109, 'Ivy', 'Clark', 'ivy.clark@company.com', '2021-08-30', 2700.00, 9),  
(110, 'Jake', 'White', 'jake.white@company.com', '2022-05-19', 3600.00, 7);  
  
-- Projects  
INSERT INTO projects VALUES (201, 'HR Revamp', '2023-01-01', '2023-12-31');  
INSERT INTO projects VALUES (202, 'Finance Automation', '2022-05-15', '2023-04-30');  
INSERT INTO projects VALUES (203, 'IT Infrastructure Upgrade', '2024-01-01', NULL);  
INSERT INTO projects VALUES (204, 'Marketing Blitz 2025', '2025-02-01', '2025-06-30');  
INSERT INTO projects VALUES (205, 'Legal Compliance', '2023-07-10', '2024-01-10');  
INSERT INTO projects VALUES (206, 'Customer Portal', '2021-11-01', '2022-10-31');  
INSERT INTO projects VALUES (207, 'Sales Booster', '2022-04-01', '2023-03-31');  
INSERT INTO projects VALUES (208, 'R&D Pilot', '2025-01-01', NULL);  
INSERT INTO projects VALUES (209, 'Procurement Tracker', '2024-03-15', '2024-11-15');  
INSERT INTO projects VALUES (210, 'Operations Streamline', '2022-09-01', '2023-09-01');  
  
-- Employee\_Projects  
INSERT INTO employee projects VALUES (101, 201, '2023-01-10');

INSERT INTO employee projects VALUES (102, 203, '2024-01-05');

INSERT INTO employee projects VALUES (103, 202, '2022-05-20');

INSERT INTO employee projects VALUES (104, 204, '2025-02-10');

INSERT INTO employee projects VALUES (105, 203, '2024-01-07');

INSERT INTO employee projects VALUES (106, 207, '2022-04-15');

INSERT INTO employee projects VALUES (107, 205, '2023-07-15');

INSERT INTO employee projects VALUES (108, 210, '2022-09-10');

INSERT INTO employee projects VALUES (109, 208, '2025-01-10');

INSERT INTO employee projects VALUES (110, 206, '2021-11-05');

# 4. 50 SQL Exercises (Questions)

**1. Concatenate first and last name as full \_name**

SELECT CONCAT(first \_name, ' ', last \_name) AS full \_name FROM employees;

**2. Convert all employee names to lowercase**

SELECT LOWER(first \_name) AS first \_name, LOWER(last \_name) AS last \_name FROM employees;

**3. Extract first 3 letters of the employee's first name**

SELECT SUBSTRING (first\_ name, 1, 3) AS short \_name FROM employees;

**4. Replace '@company.com' in email with '@org.com'**

SELECT REPLACE (email, '@company.com', '@org.com') AS new \_email FROM employees;

**5. Trim spaces from a padded string (example)**

SELECT TRIM (' HR Department ') AS trimmed;

**6. Count characters in full name**

SELECT LENGTH (CONCAT (first \_name, ' ', last \_name)) AS name \_length FROM employees;

**7. Find position of '@' in email**

SELECT INSTR(email, '@') AS position \_of \_at FROM employees;

**8. Add Mr. /Ms. before names (assuming gender column exists)**

SELECT CASE WHEN gender = 'Male' THEN CONCAT('Mr. ', first \_name, ' ', last \_name)

WHEN gender = 'Female' THEN CONCAT ('Ms. ', first \_name, ' ', last \_name) END AS titled \_name FROM employees;

**9. Format project names to uppercase**

SELECT UPPER (project \_name) AS upper \_name FROM projects;

**10. Remove dashes from project names**

SELECT REPLACE (project \_name, '-', '') AS cleaned \_name FROM projects;

**11. Label like “Emp: John Doe (HR)”**

SELECT CONCAT('Emp: ', first\_name, ' ', last\_name, ' (', department\_name, ')') AS label

FROM employees JOIN departments USING (department\_id);

**12. Check email length**

SELECT LENGTH (email) AS email \_length FROM employees;

**13. Extract last name from email (before @)**

SELECT SUBSTRING \_ INDEX (SUBSTRING\_INDEX(email, '@', 1), '.', -1) AS last \_name FROM employees;

**14.Format: “LASTNAME, Firstname”**

SELECT CONCAT (UPPER(last \_name), ', ', first \_name) AS formatted \_name FROM employees;

**15. Add “(Active)” if assigned to project with no end date**

SELECT CONCAT (first \_name, ' ', last \_name, CASE WHEN EXISTS ( SELECT 1 FROM employee \_projects ep JOIN projects p ON ep. Project \_id = p.project \_id WHERE ep.employee \_id = employees .employee \_id AND p. end \_date IS NULL ) THEN ' (Active)'ELSE ''END ) AS employee \_status FROM employees;

**16. Round salary**

SELECT salary, ROUND(salary, 0) AS rounded FROM employees;

**17. Even salaries only**

SELECT \* FROM employees WHERE MOD(salary, 2) = 0;

**18. Difference between project end and start date**

SELECT project \_name, DATEDIFF (end \_date, start \_date) AS duration \_days FROM projects WHERE end \_date IS NOT NULL;

**19. Absolute salary difference between two employees**

SELECT ABS(e1.salary - e2.salary) AS difference

FROM employees e1, employees e2

WHERE e1.employee\_id = 101 AND e2.employee\_id = 102;

**20. Raise salary by 10%**

SELECT employee \_id, salary, ROUND (salary \* POWER(1.10, 1), 2) AS raised\_ salary FROM employees;

**21. Random number**

SELECT RAND () AS random \_number;

**22. Use CEIL and FLOOR on salary**

SELECT salary, CEIL(salary) AS ceiling, FLOOR(salary) AS floor FROM employees;

**23. LENGTH of phone number (assuming column exists)**

SELECT LENGTH(phone) FROM employees;

**24. Salary category**

SELECT salary,

CASE

WHEN salary >= 5000 THEN 'High'

WHEN salary >= 3000 THEN 'Medium'

ELSE 'Low'

END AS category

FROM employees;

**25. Count digits in salary**

SELECT salary, LENGTH(FLOOR(salary)) AS digit \_count FROM employees;

**26. Today's date**

SELECT CURRENT\_DATE AS today;

**27. How many days an employee has worked**

SELECT employee \_id, DATEDIFF(CURRENT\_DATE, hire \_date) AS days \_worked FROM employees;

**28. Employees hired this year**

SELECT \* FROM employees WHERE YEAR(hire \_date) = YEAR(CURDATE());

**29. Current date and time**

SELECT NOW() AS current \_datetime;

**30. Extract year, month, and day**

SELECT hire \_date, YEAR(hire \_date) AS year, MONTH (hire \_date) AS month, DAY(hire \_date) AS day FROM employees;

**31. Employees hired before 2020**

SELECT \* FROM employees WHERE hire \_date < '2020-01-01';

**32. Projects ended in last 30 days**

SELECT \* FROM projects WHERE end \_date BETWEEN CURDATE() - INTERVAL 30 DAY AND CURDATE();

**33. Total days between start and end**

SELECT project \_name, DATEDIFF(end \_date, start \_date) AS total\_ days FROM projects WHERE end \_date IS NOT NULL;

**34. Format date: '2025-07-23' → 'July 23, 2025'**

SELECT DATE\_FORMAT('2025-07-23', '%M %d, %Y') AS formatted;

**35. Project status**

SELECT project \_name,

CASE

WHEN end\_ date IS NULL THEN 'Ongoing'

ELSE 'Completed'

END AS status

FROM projects;

**36. CASE to label salaries**

SELECT salary,

CASE

WHEN salary >= 5000 THEN 'High'

WHEN salary >= 3000 THEN 'Medium'

ELSE 'Low'

END AS label

FROM employees;

**37. COALESCE to show 'No Email'**

SELECT COALESCE (email, 'No Email') AS safe\_ email FROM employees;

**38. If hire \_date < 2015, mark as ‘Veteran’**

SELECT employee \_id,

CASE

WHEN hire \_date < '2015-01-01' THEN 'Veteran'

ELSE 'New'

END AS status

FROM employees;

**39. Default salary = 3000 if NULL**

SELECT employee\_ id, COALESCE( salary, 3000) AS salary FROM employees;

**40. Categorize department**

SELECT department \_name,

CASE

WHEN department\_ name = 'Information Technology' THEN 'IT'

WHEN department\_ name = 'Human Resources' THEN 'HR'

ELSE 'Other'

END AS category

FROM departments;

**41. Employees without project**

SELECT e. employee \_id, first \_name,

CASE

WHEN ep. project \_id IS NULL THEN 'Unassigned'

ELSE 'Assigned'

END AS assignment

FROM employees e

LEFT JOIN employee \_projects ep ON e .employee\_ id = ep. employee\_ id;

**42. Tax band**

SELECT salary,

CASE

WHEN salary >= 6000 THEN 'Band A'

WHEN salary >= 4000 THEN 'Band B'

ELSE 'Band C'

END AS tax \_band

FROM employees;

**43. Project duration**

SELECT project \_name,

CASE

WHEN end \_date IS NULL THEN 'Ongoing'

WHEN DATEDIFF (end\_ date, start \_date) > 365 THEN 'Long Term'

ELSE 'Short Term'

END AS duration

FROM projects;

**44. Even or Odd ID**

SELECT employee \_id, CASE WHEN MOD(employee\_ id, 2) = 0 THEN 'Even' ELSE 'Odd' END AS parity FROM employees;

**45. Fallback name**

SELECT COALESCE (CONCAT (first \_name, ' ', last \_name), 'Unnamed') AS name FROM employees;

**46. Label if name is long**

SELECT first\_ name, last \_name, CASE WHEN LENGTH(first \_name) + LENGTH(last \_name) > 10 THEN 'Long Name' ELSE 'Normal' END AS label FROM employees;

**47. Dummy account if email has 'TEST'**

SELECT email, CASE WHEN UPPER(email) LIKE '%TEST%' THEN 'Dummy' ELSE 'Real' END AS account \_type FROM employees;

**48. Seniority by hire year**

SELECT employee \_id, CASE WHEN YEAR(hire \_date) <= 2015 THEN 'Senior' WHEN YEAR(hire \_date) <= 2020 THEN 'Mid' ELSE 'Junior' END AS seniority FROM employees;

**49. Salary increment plan**

SELECT salary, CASE WHEN salary < 3000 THEN 'Increase by 20%' WHEN salary < 5000 THEN 'Increase by 10%' ELSE 'No Increase END AS increment \_plan FROM employees;

**50. Anniversary this month?**

SELECT employee \_id, hire \_date, CASE WHEN MONTH (hire \_date) = MONTH (CURDATE()) THEN 'Anniversary Month' ELSE 'Not Anniversary’ END AS anniversary \_statusFROM employees;

**THANK YOU!!!!!!!!!!!**