NAMES: Nsengiyumva Isaro Aurore Armelle

ID : 27882

GROUP : E

Database Management systems correction

We created a Database called employees_management, by using this query: CREATE DATABASE employees_management;
We had to connect to the database for our next queries to work: \c employees_management;
We created all tables and we inserted all values:
1.CREATE TABLE departments (

  department_id INT PRIMARY KEY,

  department_name VARCHAR(100));
2. 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));
3. CREATE TABLE projects (

  project_id INT PRIMARY KEY,

  project_name VARCHAR(100),

  start_date DATE,

  end_date DATE);
4. 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),

```sql
  FOREIGN KEY (project_id) REFERENCES projects(project_id));
```
Inserting data:
```sql
INSERT INTO departments (department_id, department_name) VALUES

(1, 'Human Resources'),

(2, 'Finance'),

(3, 'Information Technology'),

(4, 'Marketing'),

(5, 'Legal'),

(6, 'Operations'),

(7, 'Customer Service'),

(8, 'Sales'),

(9, 'Research and Development'),

(10, 'Procurement');
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);
INSERT INTO projects (project_id, project_name, start_date, end_date) VALUES

(201, 'HR Revamp', '2023-01-01', '2023-12-31'),

(202, 'Finance Automation', '2022-05-15', '2023-04-30'),

(203, 'IT Infrastructure Upgrade', '2024-01-01', NULL),

(204, 'Marketing Blitz 2025', '2025-02-01', '2025-06-30'),
```

(205, 'Legal Compliance', '2023-07-10', '2024-01-10'),

(206, 'Customer Portal', '2021-11-01', '2022-10-31'),

(207, 'Sales Booster', '2022-04-01', '2023-03-31'),

(208, 'R&D Pilot', '2025-01-01', NULL),

(209, 'Procurement Tracker', '2024-03-15', '2024-11-15'),

(210, 'Operations Streamline', '2022-09-01', '2023-09-01');
INSERT INTO employee_projects (employee_id, project_id, assigned_date) VALUES

(101, 201, '2023-01-10'),

(102, 203, '2024-01-05'),

(103, 202, '2022-05-20'),

(104, 204, '2025-02-10'),

(105, 203, '2024-01-07'),

(106, 207, '2022-04-15'),

(107, 205, '2023-07-15'),

(108, 210, '2022-09-10'),

(109, 208, '2025-01-10'),

(110, 206, '2021-11-05');

Q1. Concatenate first and last name as full_name

ANSWER: SELECT CONCAT(first_name, ' ', last_name) AS full_name FROM employees;
Q2. Convert all employee names to lowercase

ANSWER: SELECT LOWER(first_name) AS first_name_lower, LOWER(last_name) AS last_name_lower FROM employees;

Q3. Extract first 3 letters of employee's first name

ANSWER: SELECT SUBSTR(EMP_FNAME, 1, 3) AS first_3_letters

FROM EMPLOYEES;

Q4: Replace '@company.com' in email with '@org.com'

ANSWER: SELECT REPLACE(EMAIL, '@company.com', '@org.com') AS updated_email

FROM EMPLOYEES;

Q5. Trim spaces from a padded string

ANSWER: SELECT TRIM(EMP_FNAME) AS trimmed_fname

FROM EMPLOYEES;

Q6. Count characters in employee's full name

ANSWER: SELECT LENGTH(EMP_FNAME || EMP_LNAME) AS name_length

FROM EMPLOYEES;

Q7. Find position of '@' in email using INSTR

ANSWER: SELECT POSITION('@' IN email) AS at_position FROM employees;

Q8. Add 'Mr.' or 'Ms.' before names based on gender

As the gender column was not there, we will need to add it first by following this query: ALTER TABLE employees ADD COLUMN gender VARCHAR(10);
ANSWER: SELECT CASE WHEN gender = 'Male' THEN CONCAT('Mr. ', first_name, ' ', last_name)
WHEN gender = 'Female' THEN CONCAT('Ms. ', first_name, ' ', last_name) ELSE CONCAT(first_name, ' ', last_name) END AS titled_name FROM employees;

Q9: Format project names to uppercase

ANS: SELECT UPPER(project_name) AS upper_project_name FROM projects;

Q10. Remove any dashes from project names

ANS: SELECT REPLACE(project_name, '-', '') AS clean_name FROM projects;

Q11. Create a label like "Emp: John Doe (HR)"

ANS: SELECT CONCAT('Emp: ', e.first_name, ' ', e.last_name, ' (', d.department_name, ')') AS label

FROM employees e JOIN departments d ON e.department_id = d.department_id;

Q12. Check email length for each employee

ANS: SELECT email, LENGTH(email) AS email_length FROM employees;

Q13. Extract last name only from email (before @)

```
ANS: SELECT SUBSTRING(email FROM 1 FOR POSITION('@' IN email) -
1) AS name_part FROM employees;
```

Q14. Format: “LASTNAME, Firstname” using UPPER and CONCAT

ANS: SELECT CONCAT(UPPER(last_name), ', ', first_name) AS formatted_name FROM employees;

Q15. Add “(Active)” next to employee names who have current projects

ANS: SELECT CONCAT(first_name, ' ', last_name, CASE WHEN p.end_date IS NULL THEN ' (Active)' ELSE '' END) AS status_name FROM employees e LEFT JOIN employee_projects ep ON e.employee_id = ep.employee_id LEFT JOIN projects p ON ep.project_id = p.project_id;

Numeric Function Exercises (16–25)

Q16. Round salary to nearest whole number

ANS: SELECT first_name, salary, ROUND(salary) AS rounded_salary FROM employees;

Q17. Show only even salaries using MOD

ANS: SELECT first_name, salary FROM employees WHERE MOD(ROUND(salary), 2) = 0;

Q18. Show difference between two project end/start dates using DATEDIFF

ANSW: SELECT project_name, end_date - start_date AS duration_days FROM projects WHERE end_date IS NOT NULL;

Q19. Show absolute difference in salaries between two employees

ANS: SELECT ABS(e1.salary - e2.salary) AS salary_diff FROM employees e1 JOIN employees e2 ON e1.employee_id = 101 AND e2.employee_id = 103;

Q20. Raise salary by 10% using POWER

ANS: SELECT first_name, salary, salary * POWER(1.1, 1) AS increased_salary FROM employees;

Q21. Generate a random number for testing IDs
ANS: SELECT employee_id, RANDOM() AS test_random FROM employees;

Q22. Use CEIL and FLOOR on a floating salary
ANS: SELECT first_name, salary, CEIL(salary) AS ceil_salary, FLOOR(salary) AS floor_salary FROM employees;

Q23. Use LENGTH() on phone numbers

First let's add the column phone temporarily

```
ALTER TABLE employees ADD COLUMN phone VARCHAR(20);
SELECT first_name, phone, LENGTH(phone) AS phone_length FROM employees;
```

Q24. Count digits in salary amount

ANS: SELECT first_name, salary, CASE WHEN salary >= 5000 THEN 'High' WHEN salary >= 3000 THEN 'Medium' ELSE 'Low' END AS salary_category FROM employees;

Q25. Categorize salary: High/Medium/Low using CASE
ANS: SELECT first_name, salary, LENGTH(REPLACE(salary::TEXT, '.', '')) AS digit_count FROM employees;

Date/Time Function Exercises (26–35)

Q26. Show today's date
ANS: SELECT CURRENT_DATE AS today_date;

Q27. Calculate how many days an employee has worked

ANS: SELECT first_name, hire_date, CURRENT_DATE - hire_date AS days_worked

FROM employees;

Q28. Show employees hired in the current year

ANS: SELECT * FROM employees WHERE EXTRACT(YEAR FROM hire_date) = EXTRACT(YEAR FROM CURRENT_DATE);

Q29. Display current date and time

ANS: SELECT NOW() AS current_datetime;

Q30. Extract year, month, day from hire_date

ANS: SELECT first_name, hire_date, EXTRACT(YEAR FROM hire_date) AS hire_year, EXTRACT(MONTH FROM hire_date) AS hire_month, EXTRACT(DAY FROM hire_date) AS hire_day

FROM employees;

Q31. Show employees hired before 2020

ANS: SELECT * FROM employees WHERE hire_date < '2020-01-01';

Q32. List projects that ended in last 30 days
ANS: SELECT * FROM projects WHERE end_date IS NOT NULL AND end_date >= CURRENT_DATE - INTERVAL '30 days' AND end_date <= CURRENT_DATE;

Q33. Calculate total days between project start and end
ANS: SELECT project_name, end_date - start_date AS duration_days FROM projects WHERE end_date IS NOT NULL;

Q34. Format date '2025-07-23' to 'July 23, 2025'
ANS: SELECT CONCAT(TO_CHAR(DATE '2025-07-23', 'Month'), TO_CHAR(DATE '2025-07-23', 'DD, YYYY')) AS formatted_date;
Use trim to clean up spaces, SELECT CONCAT(TRIM(TO_CHAR(DATE '2025-07-23', 'Month')), TO_CHAR(DATE '2025-07-23', ' DD, YYYY')) AS formatted_date;

Q35. Add CASE: if project active (end_date IS NULL), show 'Ongoing'
ANS: SELECT project_name, CASE WHEN end_date IS NULL THEN 'Ongoing' ELSE 'Completed' END AS project_status FROM projects;

Conditional Function Exercises (36–50)

Q36. Use CASE to label salaries
ANS: SELECT first_name, salary, CASE WHEN salary >= 6000 THEN 'Very High' WHEN salary >= 5000 THEN 'High' WHEN salary >= 3000 THEN 'Medium' ELSE 'Low' END AS salary_label FROM employees;

Q37. Use COALESCE to show 'No Email' if email is NULL
UPDATE employees SET email = NULL WHERE employee_id = 110;
ANS: SELECT first_name, COALESCE(email, 'No Email') AS email_display FROM employees;

Q38. CASE: If hire_date < 2015, mark as 'Veteran'
ANS: SELECT first_name, hire_date, CASE WHEN hire_date < '2015-01-01' THEN 'Veteran' ELSE 'Newer' END AS status FROM employees;

Q39. If salary is NULL, default to 3000 using COALESCE
ANS: SELECT first_name, COALESCE(salary, 3000) AS updated_salary FROM employees;
40. Categorize departments (IT, HR, Other)
ANS: SELECT d.department_name, CASE WHEN d.department_name = 'Information Technology' THEN 'IT' WHEN d.department_name = 'Human Resources' THEN 'HR' ELSE 'Other' END AS category FROM departments d;

Q41. CASE: If employee has no project, mark as 'Unassigned'
ANS: SELECT e.first_name, CASE WHEN ep.employee_id IS NULL THEN 'Unassigned' ELSE 'Assigned' END AS project_status FROM employees e LEFT JOIN employee_projects ep ON e.employee_id = ep.employee_id;

Q42. Show tax band based on salary
ANS: SELECT first_name, salary, CASE WHEN salary >= 6000 THEN 'Band A' WHEN salary >= 4000 THEN 'Band B' ELSE 'Band C' END AS tax_band FROM employees;

Q43. Nested CASE to label project duration
ANS: SELECT project_name, end_date - start_date AS duration, CASE WHEN end_date IS NULL THEN 'Ongoing' WHEN end_date - start_date > 300 THEN 'Long-Term' WHEN end_date - start_date > 100 THEN 'Mid-Term' ELSE 'Short-Term' END AS duration_label FROM projects;

Q44. CASE with MOD to show even/odd salary IDs

ANS: SELECT employee_id, CASE WHEN MOD(employee_id, 2) = 0 THEN 'Even' ELSE 'Odd' END AS id_parity FROM employees;

Q45. Combine COALESCE + CONCAT for fallback names
ANS: SELECT employee_id, CONCAT(COALESCE(first_name, 'Unknown'), ' ' , COALESCE(last_name, 'Employee')) AS full_name

FROM employees;

Q46. CASE with LENGTH(): if name length > 10, label "Long Name"

ANSWER: SELECT first_name, last_name, CASE WHEN LENGTH(first_name || last_name) > 10 THEN 'Long Name' ELSE 'Normal' END AS name_length_status FROM employees;

Q47. CASE + UPPER(): if email has 'TEST', mark as dummy account

ANS:SELECT email, CASE WHEN UPPER(email) LIKE '%TEST%' THEN 'Dummy' ELSE 'Real' END AS email_type FROM employees;

Q48. Show seniority based on hire year
ANS:SELECT first_name, hire_date, CASE WHEN EXTRACT(YEAR FROM hire_date) <= 2015 THEN 'Senior' WHEN EXTRACT(YEAR FROM hire_date) <= 2020 THEN 'Mid-level' ELSE 'Junior' END AS seniority FROM employees;

Q49. CASE to determine salary increment range

ANSWER: SELECT first_name, salary, CASE WHEN salary < 3000 THEN 'Increase by 20%' WHEN salary < 5000 THEN 'Increase by 15%' ELSE 'Increase by 10%' END AS increment_plan FROM employees;

Q50 Use CASE with CURDATE() to determine anniversary month

ANS: SELECT first_name, hire_date, CASE WHEN EXTRACT(MONTH FROM hire_date = EXTRACT(MONTH FROM CURRENT_DATE) THEN 'Anniversary Month' ELSE 'Not Anniversary Month' END AS anniversary_status FROM employees;

END