Name: Niyobuhungiro Divin

Student ID: 28516

DBMS ASSIGNMENT:

I. String Function:

1. Concatenate first and last name as full_name:

```
employees_management=# select concat(first_name, ' ', Last_name) as Full_name from employees;
   full_name
---------------
 Alice Johnson
 Bob Smith
 Carol Adams
 David Lee
 Eve Martins
 Frank Green
 Grace Brown
 Hank Wilson
 Ivy Clark
 Jake White
(10 rows)
```

2. Convert all employee names to lowercase.

```
employees_management=# select lower(first_name), lower(last_name) from employees;
 lower |   lower
-------+---------
 alice | johnson
 bob   | smith
 carol | adams
 david | lee
 eve   | martins
 frank | green
 grace | brown
 hank  | wilson
 ivy   | clark
 jake  | white
(10 rows)
```

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

```
employees_management=# SELECT LEFT(first_name, 3) FROM employees;
 left
------
 Ali
 Bob
 Car
 Dav
 Eve
 Fra
 Gra
 Han
 Ivy
 Jak
(10 rows)
```

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

```
employees_management=# SELECT REPLACE(email, '@company.com', '@org.com') AS updated_email FROM employees;
     updated_email
-----------------------
 alice.johnson@org.com
 bob.smith@org.com
 carol.adams@org.com
 david.lee@org.com
 eve.martins@org.com
 frank.green@org.com
 grace.brown@org.com
 hank.wilson@org.com
 ivy.clark@org.com
 jake.white@org.com
(10 rows)
```

5. Trim spaces from a padded string. For instance; All columns that have datatype string, we can do:

```
employees_management=# SELECT employee_id, TRIM(first_name) AS trimmed_first_name, TRIM(last_name) AS trimmed_last_name, TRIM(email) AS trimmed_email FROM employees;
 employee_id | trimmed_first_name | trimmed_last_name |      trimmed_email
-------------+--------------------+-------------------+--------------------------
         101 | Alice              | Johnson           | alice.johnson@company.com
         102 | Bob                | Smith             | bob.smith@company.com
         103 | Carol              | Adams             | carol.adams@company.com
         104 | David              | Lee               | david.lee@company.com
         105 | Eve                | Martins           | eve.martins@company.com
         106 | Frank              | Green             | frank.green@company.com
         107 | Grace              | Brown             | grace.brown@company.com
         108 | Hank               | Wilson            | hank.wilson@company.com
         109 | Ivy                | Clark             | ivy.clark@company.com
         110 | Jake               | White             | jake.white@company.com
(10 rows)
```

6. Count characters in an employee's full name.

```
employees_management=# SELECT LENGTH(CONCAT(first_name, ' ', last_name)) AS name_length FROM employees;
 name_length
-------------
          13
           9
          11
           9
          11
          11
          11
          11
           9
          10
(10 rows)
```

7. Find position of '@' in email using INSTR()/CHARINDEX().

```
employees_management=# SELECT POSITION('@' IN email) AS at_position FROM employees;
 at_position
-------------
          14
          10
          12
          10
          12
          12
          12
          12
          10
          11
(10 rows)
```

8. Add 'Mr.' or 'Ms.' before names based on gender (assume gender exists).

```
employees_management=# SELECT CASE gender WHEN 'M' THEN CONCAT('Mr. ', first_name, ' ', last_name) WHEN 'F' THEN CONCAT('Ms. ', first_name, ' ', last_name) END AS titled_na
me FROM employees;
ERROR:  column "gender" does not exist
LINE 1: SELECT CASE gender WHEN 'M' THEN CONCAT('Mr. ', first_name, ...
                    ^
employees_management=#
```

Or without having that column:

```
employees_management=# SELECT CASE  WHEN first_name IN ('Alice', 'Carol', 'Eve', 'Grace', 'Ivy') THEN CONCAT('Ms. ', first_name, ' ', last_name)  ELSE CONCAT('Mr. ', first_
name, ' ', last_name)  END AS titled_name  FROM employees;
    titled_name
------------------
 Ms. Alice Johnson
 Mr. Bob Smith
 Ms. Carol Adams
 Mr. David Lee
 Ms. Eve Martins
 Mr. Frank Green
 Ms. Grace Brown
 Mr. Hank Wilson
 Ms. Ivy Clark
 Mr. Jake White
(10 rows)
```

9. Format project names to uppercase.

```
employees_management=# SELECT UPPER(project_name) FROM projects;
            upper
---------------------------
 HR REVAMP
 FINANCE AUTOMATION
 IT INFRASTRUCTURE UPGRADE
 MARKETING BLITZ 2025
 LEGAL COMPLIANCE
 CUSTOMER PORTAL
 SALES BOOSTER
 R&D PILOT
 PROCUREMENT TRACKER
 OPERATIONS STREAMLINE
(10 rows)
```

10. Remove any dashes from project names.

```
employees_management=# SELECT REPLACE(project_name, '-', '') FROM projects;
          replace
--------------------------
 HR Revamp
 Finance Automation
 IT Infrastructure Upgrade
 Marketing Blitz 2025
 Legal Compliance
 Customer Portal
 Sales Booster
 R&D Pilot
 Procurement Tracker
 Operations Streamline
(10 rows)
```

11. Create a label like “Emp: John Doe (HR)”.

```
employees_management=# SELECT CONCAT('Emp: ', first_name, ' ', last_name, ' (', department_name, ')') AS label  FROM employees e JOIN departments d ON e.department_id = d.d
epartment_id;
               label
-----------------------------------------
 Emp: Alice Johnson (Human Resources)
 Emp: Bob Smith (Information Technology)
 Emp: Carol Adams (Finance)
 Emp: David Lee (Marketing)
 Emp: Eve Martins (Information Technology)
 Emp: Frank Green (Sales)
 Emp: Grace Brown (Legal)
 Emp: Hank Wilson (Operations)
 Emp: Ivy Clark (Research and Development)
 Emp: Jake White (Customer Service)
(10 rows)
```

12. Check email length for each employee.

```
employees_management=# SELECT email, LENGTH(email) AS email_length FROM employees;
           email            | email_length
----------------------------+--------------
 alice.johnson@company.com  |           25
 bob.smith@company.com      |           21
 carol.adams@company.com    |           23
 david.lee@company.com      |           21
 eve.martins@company.com    |           23
 frank.green@company.com    |           23
 grace.brown@company.com    |           23
 hank.wilson@company.com    |           23
 ivy.clark@company.com      |           21
 jake.white@company.com     |           22
(10 rows)
```

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

```
employees_management=# SELECT SPLIT_PART(email, '@', 1) AS email_user FROM employees;
  email_user
---------------
 alice.johnson
 bob.smith
 carol.adams
 david.lee
 eve.martins
 frank.green
 grace.brown
 hank.wilson
 ivy.clark
 jake.white
(10 rows)
```

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

```
employees_management=# SELECT CONCAT(UPPER(last_name), ', ', first_name) AS formatted_name FROM employees;
 formatted_name
---------------
 JOHNSON, Alice
 SMITH, Bob
 ADAMS, Carol
 LEE, David
 MARTINS, Eve
 GREEN, Frank
 BROWN, Grace
 WILSON, Hank
 CLARK, Ivy
 WHITE, Jake
(10 rows)
```

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

```
employees_management=# SELECT e.first_name, e.last_name, CASE WHEN p.end_date IS NULL OR p.end_date > CURRENT_DATE THEN CONCAT(e.first_name, ' ', e.last_name, ' (Active)')
ELSE CONCAT(e.first_name, ' ', e.last_name) END AS status FROM employees e LEFT JOIN employee_projects ep ON e.employee_id = ep.employee_id LEFT JOIN projects p ON ep.proje
ct_id = p.project_id;
 first_name | last_name |       status
------------+-----------+--------------------
 Alice      | Johnson   | Alice Johnson
 Bob        | Smith     | Bob Smith (Active)
 Carol      | Adams     | Carol Adams
 David      | Lee       | David Lee
 Eve        | Martins   | Eve Martins (Active)
 Frank      | Green     | Frank Green
 Grace      | Brown     | Grace Brown
 Hank       | Wilson    | Hank Wilson
 Ivy        | Clark     | Ivy Clark (Active)
 Jake       | White     | Jake White
(10 rows)
```

II. Numeric Function:

16. Round salary to the nearest whole number.

```
employees_management=# SELECT salary, ROUND(salary) AS rounded_salary FROM employees;
 salary  | rounded_salary
---------+----------------
 4500.00 |           4500
 5200.00 |           5200
 6700.00 |           6700
 3800.00 |           3800
 4000.00 |           4000
 6000.00 |           6000
 4900.00 |           4900
 3100.00 |           3100
 2700.00 |           2700
 3600.00 |           3600
(10 rows)
```

17. Show only even salaries using MOD.

```
employees_management=# SELECT * FROM employees WHERE MOD(salary, 2) = 0;
 employee_id | first_name | last_name |           email           | hire_date  | salary  | department_id
-------------+------------+-----------+---------------------------+------------+---------+---------------
         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
(10 rows)
```

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

```
employees_management=# SELECT project_name, (end_date - start_date) AS duration_days  FROM projects  WHERE end_date IS NOT NULL;
     project_name      | duration_days
-----------------------+---------------
 HR Revamp             |           364
 Finance Automation    |           350
 Marketing Blitz 2025  |           149
 Legal Compliance      |           184
 Customer Portal       |           364
 Sales Booster         |           364
 Procurement Tracker   |           245
 Operations Streamline |           365
(8 rows)
```

19. Show absolute difference in salaries between two employees.

```
employees_management=# SELECT ABS(e1.salary - e2.salary) AS salary_diff FROM employees e1  JOIN employees e2 ON e1.employee_id = 101 AND e2.employee_id = 102;
 salary_diff
-------------
      700.00
(1 row)
```

20. Raise salary by 10% using POWER.

```
employees_management=# SELECT salary, salary * POWER(1.10, 1) AS raised_salary FROM employees;
 salary  |       raised_salary
---------+---------------------------
 4500.00 | 4950.000000000000000000
 5200.00 | 5720.000000000000000000
 6700.00 | 7370.000000000000000000
 3800.00 | 4180.000000000000000000
 4000.00 | 4400.000000000000000000
 6000.00 | 6600.000000000000000000
 4900.00 | 5390.000000000000000000
 3100.00 | 3410.000000000000000000
 2700.00 | 2970.000000000000000000
 3600.00 | 3960.000000000000000000
(10 rows)
```

21. Generate a random number for testing IDs.

```
employees_management=# SELECT employee_id, ROUND(RANDOM() * 10000) AS random_id FROM employees;
 employee_id | random_id
-------------+-----------
         101 |       514
         102 |      8377
         103 |      1082
         104 |      4579
         105 |        70
         106 |      3488
         107 |      9852
         108 |      8646
         109 |      2869
         110 |      1228
(10 rows)
```

22. Use CEIL and FLOOR on a floating salary.

```
employees_management=# SELECT salary, CEIL(salary) AS ceil_val, FLOOR(salary) AS floor_val FROM employees;
 salary  | ceil_val | floor_val
---------+----------+-----------
 4500.00 |     4500 |      4500
 5200.00 |     5200 |      5200
 6700.00 |     6700 |      6700
 3800.00 |     3800 |      3800
 4000.00 |     4000 |      4000
 6000.00 |     6000 |      6000
 4900.00 |     4900 |      4900
 3100.00 |     3100 |      3100
 2700.00 |     2700 |      2700
 3600.00 |     3600 |      3600
(10 rows)
```

23. Use LENGTH() on phone numbers (assume column exists).

```
employees_management=# SELECT phone_number, LENGTH(phone_number) AS phone_length FROM employees;
ERROR:  column "phone_number" does not exist
LINE 1: SELECT phone_number, LENGTH(phone_number) AS phone_length FR...
               ^
employees_management=#
```

24. Categorize salary: High/Medium/Low using CASE.

```
employees_management=# SELECT salary, CASE WHEN salary >= 6000 THEN 'High' WHEN salary >= 4000 THEN 'Medium' ELSE 'Low' END AS salary_level FROM employees;
 salary  | salary_level
---------+--------------
 4500.00 | Medium
 5200.00 | Medium
 6700.00 | High
 3800.00 | Low
 4000.00 | Medium
 6000.00 | High
 4900.00 | Medium
 3100.00 | Low
 2700.00 | Low
 3600.00 | Low
(10 rows)
```

25. Count digits in salary amount.

```
employees_management=# SELECT salary, LENGTH(FLOOR(salary)::TEXT) AS digit_count FROM employees;
 salary  | digit_count
---------+-------------
 4500.00 |           4
 5200.00 |           4
 6700.00 |           4
 3800.00 |           4
 4000.00 |           4
 6000.00 |           4
 4900.00 |           4
 3100.00 |           4
 2700.00 |           4
 3600.00 |           4
(10 rows)
```

III. Date/Time Function:

26. Show today's date using CURRENT_DATE.

```
employees_management=# SELECT CURRENT_DATE;
 current_date
--------------
 2025-07-30
(1 row)
```

27. Calculate how many days an employee has worked.

```
employees_management=# SELECT first_name, CURRENT_DATE - hire_date AS days_worked FROM employees;
 first_name | days_worked
------------+-------------
 Alice      |        3790
 Bob        |        2594
 Carol      |        4706
 David      |        2033
 Eve        |        2058
 Frank      |        2944
 Grace      |        3923
 Hank       |        4546
 Ivy        |        1430
 Jake       |        1168
(10 rows)
```

28. Show employees hired in the current year.

```
employees_management=# SELECT * FROM employees WHERE EXTRACT(YEAR FROM hire_date) = EXTRACT(YEAR FROM CURRENT_DATE);
 employee_id | first_name | last_name | email | hire_date | salary | department_id
-------------+------------+-----------+-------+-----------+--------+---------------
(0 rows)


employees_management=#
```

29. Display current date and time using NOW().

```
employees_management=# SELECT NOW();
              now
--------------------------------
 2025-07-30 11:03:53.321143+02
(1 row)
```

30. Extract the year, month, and day from hire_date.

```
employees_management=# SELECT hire_date, EXTRACT(YEAR FROM hire_date) AS year, EXTRACT(MONTH FROM hire_date) AS month, EXTRACT(DAY FROM hire_date) AS day FROM employees;
 hire_date  | year | month | day
------------+------+-------+-----
 2015-03-15 | 2015 |     3 |  15
 2018-06-23 | 2018 |     6 |  23
 2012-09-10 | 2012 |     9 |  10
 2020-01-05 | 2020 |     1 |   5
 2019-12-11 | 2019 |    12 |  11
 2017-07-08 | 2017 |     7 |   8
 2014-11-02 | 2014 |    11 |   2
 2013-02-17 | 2013 |     2 |  17
 2021-08-30 | 2021 |     8 |  30
 2022-05-19 | 2022 |     5 |  19
(10 rows)
```

31. Show employees hired before 2020.

```
employees_management=# SELECT * FROM employees WHERE hire_date < '2020-01-01';
 employee_id | first_name | last_name |          email           | hire_date  | salary  | department_id
-------------+------------+-----------+--------------------------+------------+---------+---------------
         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
         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
(7 rows)
```

32. List projects that ended in the last 30 days.

```
employees_management=# SELECT * FROM projects WHERE end_date BETWEEN CURRENT_DATE - INTERVAL '30 days' AND CURRENT_DATE;
 project_id |     project_name     | start_date |  end_date
------------+----------------------+------------+------------
        204 | Marketing Blitz 2025 | 2025-02-01 | 2025-06-30
(1 row)


employees_management=#
```

33. Calculate total days between project start and end dates.

```
employees_management=# SELECT project_name, end_date - start_date AS project_length FROM projects WHERE end_date IS NOT NULL;
     project_name      | project_length
-----------------------+----------------
 HR Revamp             |            364
 Finance Automation    |            350
 Marketing Blitz 2025  |            149
 Legal Compliance      |            184
 Customer Portal       |            364
 Sales Booster         |            364
 Procurement Tracker   |            245
 Operations Streamline |            365
(8 rows)
```

34. Format date: ‘2025-07-23’ to ‘July 23, 2025’ (use CONCAT).

```
employees_management=# SELECT TO_CHAR(DATE '2025-07-23', 'Month DD, YYYY') AS formatted_date;
   formatted_date
--------------------
 July       23, 2025
(1 row)


employees_management=#
```

35. Add a CASE: if project still active (end_date IS NULL), show ‘Ongoing’.

```
employees_management=# SELECT project_name, CASE WHEN end_date IS NULL THEN 'Ongoing' ELSE 'Completed' END AS status FROM projects;
       project_name        |  status
---------------------------+-----------
 HR Revamp                 | Completed
 Finance Automation        | Completed
 IT Infrastructure Upgrade | Ongoing
 Marketing Blitz 2025      | Completed
 Legal Compliance          | Completed
 Customer Portal           | Completed
 Sales Booster             | Completed
 R&D Pilot                 | Ongoing
 Procurement Tracker       | Completed
 Operations Streamline     | Completed
(10 rows)
```

IV. Conditional Function:

36. Use CASE to label salaries.

```
employees_management=# SELECT salary, CASE WHEN salary > 6000 THEN 'Top Earner' WHEN salary BETWEEN 4000 AND 6000 THEN 'Average' ELSE 'Low Earner' END AS salary_label FROM
employees;
 salary  | salary_label
---------+--------------
 4500.00 | Average
 5200.00 | Average
 6700.00 | Top Earner
 3800.00 | Low Earner
 4000.00 | Average
 6000.00 | Average
 4900.00 | Average
 3100.00 | Low Earner
 2700.00 | Low Earner
 3600.00 | Low Earner
(10 rows)
```

37. Use COALESCE to show ‘No Email’ if email is NULL.

```
employees_management=# SELECT COALESCE(email, 'No Email') AS safe_email FROM employees;
        safe_email
---------------------------
 alice.johnson@company.com
 bob.smith@company.com
 carol.adams@company.com
 david.lee@company.com
 eve.martins@company.com
 frank.green@company.com
 grace.brown@company.com
 hank.wilson@company.com
 ivy.clark@company.com
 jake.white@company.com
(10 rows)
```

38. CASE: If hire_date < 2015, mark as ‘Veteran’.

```
employees_management=#  SELECT first_name, CASE WHEN hire_date < '2015-01-01' THEN 'Veteran' ELSE 'New' END AS status FROM employees;
 first_name | status
------------+---------
 Alice      | New
 Bob        | New
 Carol      | Veteran
 David      | New
 Eve        | New
 Frank      | New
 Grace      | Veteran
 Hank       | Veteran
 Ivy        | New
 Jake       | New
(10 rows)
```

39. If salary is NULL, default it to 3000 using COALESCE.

```
employees_management=# SELECT COALESCE(salary, 3000) AS adjusted_salary FROM employees;
 adjusted_salary
-----------------
         4500.00
         5200.00
         6700.00
         3800.00
         4000.00
         6000.00
         4900.00
         3100.00
         2700.00
         3600.00
(10 rows)
```

40. CASE: Categorize departments (IT, HR, Other).

```
employees_management=# SELECT department_name, CASE WHEN department_name = 'Information Technology' THEN 'Tech' WHEN department_name = 'Human Resources' THEN 'People' ELSE
'Other' END AS dept_type FROM departments;
     department_name      | dept_type
--------------------------+-----------
 Human Resources          | People
 Finance                  | Other
 Information Technology   | Tech
 Marketing                | Other
 Legal                    | Other
 Operations               | Other
 Customer Service         | Other
 Sales                    | Other
 Research and Development | Other
 Procurement              | Other
(10 rows)
```

41. CASE: If employee has no project, mark as ‘Unassigned’.

```
employees_management=# SELECT e.employee_id, 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;
 employee_id | first_name | project_status
-------------+------------+----------------
         101 | Alice      | Assigned
         102 | Bob        | Assigned
         103 | Carol      | Assigned
         104 | David      | Assigned
         105 | Eve        | Assigned
         106 | Frank      | Assigned
         107 | Grace      | Assigned
         108 | Hank       | Assigned
         109 | Ivy        | Assigned
         110 | Jake       | Assigned
(10 rows)
```

42. CASE: Show tax band based on salary.

```
employees_management=# SELECT salary, CASE WHEN salary > 6000 THEN '30%' WHEN salary > 4000 THEN '20%' ELSE '10%' END AS tax_band FROM employees;
  salary  | tax_band
----------+----------
 4500.00 | 20%
 5200.00 | 20%
 6700.00 | 30%
 3800.00 | 10%
 4000.00 | 10%
 6000.00 | 20%
 4900.00 | 20%
 3100.00 | 10%
 2700.00 | 10%
 3600.00 | 10%
(10 rows)
```

43. Use nested CASE to label project duration.

```
employees_management=# SELECT project_name, CASE WHEN (end_date - start_date) > 365 THEN 'Long-term' WHEN (end_date - start_date) > 180 THEN 'Medium-term' ELSE 'Short-term'
 END AS duration_type FROM projects WHERE end_date IS NOT NULL;
     project_name      | duration_type
-----------------------+---------------
 HR Revamp             | Medium-term
 Finance Automation    | Medium-term
 Marketing Blitz 2025  | Short-term
 Legal Compliance      | Medium-term
 Customer Portal       | Medium-term
 Sales Booster         | Medium-term
 Procurement Tracker   | Medium-term
 Operations Streamline | Medium-term
(8 rows)
```

44. Use CASE with MOD to show even/odd salary IDs.

```
employees_management=# SELECT employee_id, salary, CASE WHEN MOD(employee_id, 2) = 0 THEN 'Even' ELSE 'Odd' END AS id_type FROM employees;
 employee_id | salary  | id_type
-------------+---------+---------
         101 | 4500.00 | Odd
         102 | 5200.00 | Even
         103 | 6700.00 | Odd
         104 | 3800.00 | Even
         105 | 4000.00 | Odd
         106 | 6000.00 | Even
         107 | 4900.00 | Odd
         108 | 3100.00 | Even
         109 | 2700.00 | Odd
         110 | 3600.00 | Even
(10 rows)
```

45. Combine COALESCE + CONCAT for fallback names.

```
employees_management=# SELECT COALESCE(CONCAT(first_name, ' ', last_name), 'Unnamed') AS display_name FROM employees;
 display_name
---------------
 Alice Johnson
 Bob Smith
 Carol Adams
 David Lee
 Eve Martins
 Frank Green
 Grace Brown
 Hank Wilson
 Ivy Clark
 Jake White
(10 rows)
```

46. CASE with LENGTH(): if name length > 10, label “Long Name”.

```
employees_management=# SELECT first_name, CASE WHEN LENGTH(first_name) > 10 THEN 'Long Name' ELSE 'Normal' END AS name_type FROM employees;
 first_name | name_type
------------+-----------
 Alice      | Normal
 Bob        | Normal
 Carol      | Normal
 David      | Normal
 Eve        | Normal
 Frank      | Normal
 Grace      | Normal
 Hank       | Normal
 Ivy        | Normal
 Jake       | Normal
(10 rows)
```

47. CASE + UPPER(): if email has ‘TEST’, mark as dummy account.

```
employees_management=# SELECT email, CASE WHEN UPPER(email) LIKE '%TEST%' THEN 'Dummy' ELSE 'Real' END AS email_type FROM employees;
           email            | email_type
----------------------------+------------
 alice.johnson@company.com  | Real
 bob.smith@company.com      | Real
 carol.adams@company.com    | Real
 david.lee@company.com      | Real
 eve.martins@company.com    | Real
 frank.green@company.com    | Real
 grace.brown@company.com    | Real
 hank.wilson@company.com    | Real
 ivy.clark@company.com      | Real
 jake.white@company.com     | Real
(10 rows)
```

48. CASE: Show seniority based on hire year (e.g., Junior/Senior).

```
employees_management=# SELECT first_name, CASE WHEN EXTRACT(YEAR FROM hire_date) < 2015 THEN 'Senior' WHEN EXTRACT(YEAR FROM hire_date) >= 2020 THEN 'Junior' ELSE 'Mid-leve
l' END AS seniority FROM employees;
 first_name | seniority
------------+-----------
 Alice      | Mid-level
 Bob        | Mid-level
 Carol      | Senior
 David      | Junior
 Eve        | Mid-level
 Frank      | Mid-level
 Grace      | Senior
 Hank       | Senior
 Ivy        | Junior
 Jake       | Junior
(10 rows)
```

49. Use CASE to determine salary increment range.

```
employees_management=# SELECT salary, CASE WHEN salary < 4000 THEN 'Raise by 20%' WHEN salary < 6000 THEN 'Raise by 10%' ELSE 'No raise' END AS increment_plan FROM employee
s;
 salary  | increment_plan
---------+---------------
 4500.00 | Raise by 10%
 5200.00 | Raise by 10%
 6700.00 | No raise
 3800.00 | Raise by 20%
 4000.00 | Raise by 10%
 6000.00 | No raise
 4900.00 | Raise by 10%
 3100.00 | Raise by 20%
 2700.00 | Raise by 20%
 3600.00 | Raise by 20%
(10 rows)
```

50. Use CASE with CURDATE() to determine anniversary month.

```
employees_management=# SELECT first_name, CASE WHEN EXTRACT(MONTH FROM hire_date) = EXTRACT(MONTH FROM CURRENT_DATE) THEN 'Anniversary Month' ELSE 'Not Anniversary' END AS
anniversary_status FROM employees;
 first_name | anniversary_status
------------+--------------------
 Alice      | Not Anniversary
 Bob        | Not Anniversary
 Carol      | Not Anniversary
 David      | Not Anniversary
 Eve        | Not Anniversary
 Frank      | Anniversary Month
 Grace      | Not Anniversary
 Hank       | Not Anniversary
 Ivy        | Not Anniversary
 Jake       | Not Anniversary
(10 rows)
```