NAMES: Murenzi Dan

ID: 28234

DBMS Assignment on functions

STRING FUNCTIONS

1. Concatenate first and last name as full_name.

select concat(first_name,' ', last_name)from employee;

2. Convert all employee names to lowercase

select lower(concat(first_name,' ', last_name)) AS full_name from employee;

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

select substring(first_name, 1,3) from employee;

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

select replace(email, '@company.com', '@org.com') from employee;

5. Trim spaces from a padded string.

select trim(first_name) from employee;

6. Count characters in an employee's full name

select length(concat(first_name,'',last_name)) from employee;

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

select instr(email, '@') as postion from employee;

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

select first_name, last_name, case when first_name in ('bob','carol','david','frank','hank','jake') then concat('Mr.',first_name,' ',last_name) else concat('Mrs.', first_name,' ',last_name) end as full_name from employee;

9. Format project names to uppercase.

select upper(project_name) from projects;

10. Remove any dashes from project names.

select project_name, replace(project_name,' ','') as cleaned_project_name from projects;

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

select concat('emp:', first_name,' ',last_name,'(HR)') as full_name from employee;

12. Check email length for each employees

select first_name, email, length(email) from employee;

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

select email, substring_index(substring_index(email,'@',1),'.',-1) as last_name_email from employee;

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

select concat(upper(last_name),' ',first_name) from employee;

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

SELECT CONCAT(first_name, ' ', last_name, CASE WHEN p.end_date IS NULL OR p.end_date >= CURDATE() THEN ' (Active)' ELSE '' END) AS employee_status FROM Employee 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 FUNCTIONS

16. Round salary to the nearest whole number

select first_name, last_name, salary, round(salary) as rounded_salary from employee;

17. Show only even salaries using MOD.

select * from employee where mod(salary, 2)=0;

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

select project_name,start_date,end_date, datediff(end_date, start_date) from projects;

***19. Show absolute difference in salaries between two employees*.**

SELECT ABS(e1.salary - e2.salary) AS salary_difference FROM Employee e1 JOIN Employees e2 ON e1.employee_id = 101 AND e2.employee_id = 102;

***20. Raise salary by 10% using POWER*.**

SELECT salary,   salary * POWER(1.10, 3) AS salary_after_3_raises FROM employee;

21. Generate a random number for testing IDs.

SELECT first_name, last_name, ROUND(salary * POWER(1.1, 1), 2) AS increased_salary FROM Employee;

22. Use CEIL and FLOOR on a floating salary

SELECT first_name, last_name,   salary,   CEIL(salary) AS ceiling_salary, FLOOR(salary) AS floor_salary FROM employee;

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

select length('phone') ;

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

select first_name, last_name, salary, case when salary >=5000 then 'high' when salary >= 3500 then 'medium' else 'low' end as salary_category from employee;

25. Count digits in salary amount.

SELECT salary,   LENGTH(REPLACE(salary, '.', '')) AS digit_count FROM employee;

DATE/TIME FUNCTIONS

26. Show today's date using CURRENT_DATE.

select current_date;

27. Calculate how many days an employee has worked.

select first_name, last_name, hire_date, datediff(current_date, hire_date) as days_worked from employee;

28. Show employees hired in the current year.

SELECT first_name, last_name, hire_date FROM employee WHERE YEAR(hire_date) = YEAR(CURRENT_DATE);

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

select now() as current_datetime;

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

SELECT first_name, last_name, hire_date, YEAR(hire_date) AS hire_year, MONTH(hire_date) AS hire_month, DAY(hire_date) AS hire_day FROM employee;

31. Show employees hired before 2020.

SELECT first_name, last_name,   hire_date FROM employee WHERE hire_date < '2020-01-01';

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

SELECT project_name, end_date FROM projects WHERE end_date BETWEEN DATE_SUB(CURRENT_DATE, INTERVAL 30 DAY) AND CURRENT_DATE;

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

SELECT project_name, start_date, end_date, DATEDIFF(end_date, start_date) AS total_duration_days FROM projects;

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

SELECT CONCAT(MONTHNAME('2025-07-23'), ' ', DAY('2025-07-23'), ', ', YEAR('2025-07-23')) AS formatted_date;

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

SELECT project_name, start_date, CASE WHEN end_date IS NULL THEN 'Ongoing' ELSE DATE_FORMAT(end_date, '%Y-%m-%d') END AS project_status_or_end_date FROM projects;

CONDICTIONAL FUNCTIONS

36. Use CASE to label salaries.

SELECT first_name, last_name, salary, CASE WHEN salary >= 5000 THEN 'High' WHEN salary >= 3500 THEN 'Medium' ELSE 'Low' END AS salary_label FROM employee;

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

SELECT first_name, last_name, COALESCE(email, 'No Email') AS email_address FROM employee;

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

SELECT first_name, last_name, hire_date, CASE WHEN hire_date < '2015-01-01' THEN 'Veteran' ELSE 'Newcomer' END AS employee_status FROM employee;

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

SELECT first_name, last_name, COALESCE(salary, 3000) AS salary_with_default FROM employee;

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

SELECT first_name, last_name, department, CASE WHEN department = 'IT' THEN 'IT' WHEN department = 'HR' THEN 'HR' ELSE 'Other' END AS department_category FROM employee;

42. CASE: Show tax band based on salary

SELECT first_name, last_name, salary, CASE WHEN salary >= 5000 THEN 'High Tax Band' WHEN salary >= 3500 THEN 'Mid Tax Band' WHEN salary >= 1000 THEN 'Low Tax Band' ELSE 'No Tax' END AS tax_band FROM employee;

43. Use nested CASE to label project duration

SELECT project_id, project_name, start_date, end_date, DATEDIFF(end_date, start_date) AS duration_days, CASE WHEN DATEDIFF(end_date, start_date) < 350 THEN 'Short' WHEN DATEDIFF(end_date, start_date) BETWEEN 350 AND 480 THEN CASE WHEN DATEDIFF(end_date, start_date) <= 90 THEN 'Medium' ELSE 'Moderately Long' END ELSE 'Long' END AS duration_label FROM projects;

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

SELECT employee_id, first_name, salary, CASE WHEN MOD(employee_id, 2) = 0 THEN 'Even' ELSE 'Odd' END AS id_parity FROM employee;

45. Combine COALESCE + CONCAT for fallback names.

SELECT employee_id, CONCAT(COALESCE(first_name, 'Unknown'), ' ', COALESCE(last_name, 'Unknown')) AS full_name FROM employee;

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

SELECT first_name, LENGTH(first_name) AS name_length, CASE WHEN LENGTH(first_name) > 10 THEN 'Long Name' ELSE 'Short Name' END AS name_label FROM employee;

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

SELECT email, CASE WHEN UPPER(email) LIKE '%TEST%' THEN 'Dummy Account' ELSE 'Real Account' END AS account_type FROM employee;

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

SELECT first_name, hire_date, CASE WHEN YEAR(hire_date) <= YEAR(CURDATE()) - 10 THEN 'Senior' WHEN YEAR(hire_date) <= YEAR(CURDATE()) - 5 THEN 'Mid-Level' ELSE 'Junior' END AS seniority_level FROM employee;

49. Use CASE to determine salary increment range.

SELECT employee_id, first_name, salary, CASE WHEN salary >= 5000 THEN 'Increase by 5%' WHEN salary >= 3500 THEN 'Increase by 10%' WHEN salary >= 2000 THEN 'Increase by 15%' ELSE 'Increase by 20%' END AS increment_range FROM employee;

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

```
SELECT employee_id, first_name, hire_date, CASE WHEN MONTH(hire_date) = MONTH(CURDATE())
THEN 'Anniversary Month' ELSE 'Not Anniversary Month' END AS anniversary_status FROM employee;
```