NAMES: NIYONKURU NYIRIMPETA Yabesi

ID: 27715

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 employee***

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 Employee 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;