

LAB CYCLE 3

Experiment No. 5

20MCA134 ADVANCED DBMS LAB

Familiarization of Stored Procedure, Function, Cursor and Triggers

1. Write a stored procedure to read three numbers and find the greatest among them.
2. Write a stored procedure to read two numbers and print all the numbers between them.
3. Write a stored procedure to read N and find the sum of the series $1+2+3+\dots N$
4. Write a stored procedure to read a mark and display the grade
5. Write a stored procedure to read a number and invert the given number
6. Create a procedure which will receive account_id and amount to withdraw. If the account does not exist, it will display a message. Otherwise, if the account exists, it will allow the withdrawal only if the new balance after the withdrawal is at least 1000.
7. Create a 'Customer' table with attributes customer id, name, city and credits. Write a stored procedure to display the details of a particular customer from the customer table, where name is passed as a parameter.
8. Create a stored procedure to determine membership of a particular customer based on the following credits: Above 5000 = Membership Platinum 1000 to 5000 = Gold < 1000 = silver [Use IN and OUT Parameters]
9. Create a function to accept the Id of an employee and return his salary
10. Write a function that takes employee name as parameter and returns the number of employees with this name. Use the function to update details of employees with unique names. For other cases, the program (not the function) should display error messages - "No Employee" or "Multiple employees".
11. Write a stored procedure using cursor to calculate salary of each employee. Consider an Emp_salary table have the following attributes emp_id, emp_name, no_of_working_days, designation and salary

DESIGNATION	DAILY WAGE AMOUNT
Assistant Professor	1750/day
Clerk	750/day
Programmar	1250/day

12. Write a procedure to calculate the electricity bill of all customers. Electricity board charges the following rates to domestic uses to find the consumption of energy.
 - a) For first 100 units Rs:2 per unit.

b) 101 to 200 units Rs:2.5 per unit.

c) 201 to 300 units Rs: 3 per unit.

d) Above 300 units Rs: 4 per unit

Consider the table 'Bill' with fields customer_id, name, pre_reading, cur_reading , unit, and amount.

13. Create a trigger on employee table such that whenever a row is deleted, it is moved to history table named 'Emp_history' with the same structure as employee table. 'Emp_history' will contain an additional column "Date_of_deletion" to store the date on which the row is removed. [After Delete Trigger]
14. Before insert a new record in emp_details table, create a trigger that check the column value of FIRST_NAME, LAST_NAME, JOB_ID and if there are any space(s) before or after the FIRST_NAME, LAST_NAME, TRIM () function will remove those. The value of the JOB_ID will be converted to upper cases by UPPER () function. [Before Insert Trigger]
15. Consider the following table with sample data. Create a trigger to calculate total marks, percentage and grade of the students, when marks of the subjects are updated. [After Update Trigger]

Stud_id	Name	Sub1	Sub2	Sub3	Sub4	Sub5	Total	Per_Marks	Grade
1	Steven King	0	0	0	0	0	0.0		
2	Neena Kochhar	0	0	0	0	0	0.0		
3	Alexander Hunold	0	0	0	0	0	0.0		
4	Lex De Haan	0	0	0	0	0	0.0		

For this sample calculation, the following conditions are assumed:

Total Marks (will be stored in TOTAL column) : $TOTAL = SUB1 + SUB2 + SUB3 + SUB4 + SUB5$.

Percentage of Marks (will be stored in PER_MARKS column): $PER_MARKS = (TOTAL)/5$

Grade (will be stored in GRADE column):

- If $PER_MARKS \geq 90 \rightarrow$ 'EXCELLENT'

- If $PER_MARKS \geq 75$ AND $PER_MARKS < 90 \rightarrow$ 'VERY GOOD'

- If $PER_MARKS \geq 60$ AND $PER_MARKS < 75 \rightarrow$ 'GOOD'

- If $PER_MARKS \geq 40$ AND $PER_MARKS < 60 \rightarrow$ 'AVERAGE'

- If $PER_MARKS < 40 \rightarrow$ 'NOT PROMOTED'