Subprograms and Packages

Subprograms (Procedures and Functions)

Practice 1

- Create a procedure called USER_QUERY_EMP that accepts three parameters. Parameter p_myeno is of IN parameter mode which provides the empno value. The other two parameters p_myjob and p_mysal are of OUT mode. The procedure retrieves the salary and job of an employee with the provided employee number and assigns those to the two OUT parameters respectively. The procedure should handle the error if the empno does not exist in the EMP table by displaying an appropriate message. Use bind variables for the two OUT Parameters.
- Compile the code, invoke the procedure, and display the salary and job title for employee number 7839. Do the same for employee number 7123.

Practice 2

- Create a function named USER_ANNUAL_COMP that has a parameter p_eno for passing on the values of an employee number of the employee. The function calculates and returns the annual compensation of the employee by using the following formula.
  annual_compensation = (p_sal+p_comm)*12
  If the salary or commission value is NULL then zero should be substituted for it.
- Give a call to USER_ANNUAL_COMP from a SELECT statement, against the EMP table.

Practice 3

- Create a function named USER_VALID_DEPTNO that has a single parameter p_dno to accept a department number and returns a BOOLEAN value. The function returns TRUE if the department number exists in the DEPT table else it returns FALSE.
- Create a procedure named SHOW_STRENGTH that accepts department number in a single parameter p_deptno from user. The procedure gives a call to USER_VALID_DEPTNO. If the function returns TRUE then the procedure finds out how many employees are there in the department from the EMP table and displays the same on the screen. If the function returns FALSE then the procedure displays an appropriate error message.
- Give call to SHOW_STRENGTH by passing on department number 10. Do the same for department number 76

Practice 4

- Create a procedure named SHOW_RECORDS that accepts a single parameter p_join_date. The procedure determines and displays on the screen, the details of the employees who have joined after p_join_date, in the following format.

Employees Joined after ddth, Month yyyy

EMPLOYEE NAME	JOB	SALARY	DEPARTMENT
XXXXXXXX		XXXXX99,999	99
XXXXXXXX		XXXXX99,999	99

The procedure should display appropriate message if there is no employee who joined after p_join_date .

- Give a call to SHOW_RECORDS from an anonymous PL/SQL block

Practice 5

- Create a procedure named ADD_EMPLOYEE to hire an employee. Parameters to the procedure are job, mgr, hiredate, salary, commission and deptno. Validate the following:

  a. Employee number is not taken as a parameter but is auto generated by using a SEQUENCE.
  b. Job is either 'CLERK' or 'ANALYST' or 'SALESMAN'. The input value can be entered in any case (upper or lower or initcap).
  c. Mgr is an existing employee.
  d. Hiredate is less than system date.
  e. Salary must be greater than 800
  f. Commission is not null if the job is SALESMAN. For any other job, commission may be null.
  g. Deptno must exist in the DEPT table.
  Insert the record if the above validations are met and display a message '1 row inserted'. If the row is not inserted generate an exception and handle it by displaying an appropriate message.
- Give a call to ADD_EMPLOYEE through an anonymous PL/SQL block

Practice 6

- Create a function named FIND_SAL_GRADE which accepts salary of an employee finds the corresponding salary grade from SALGRADE table and returns the grade. The function should raise an exception if the salary value does not fit in any of the salary ranges specified in the salgrade table.
- Create a procedure CALL_FIND_SAL_GRADE that does not accept any parameter. The procedure gives call to FIND_SAL_GRADE for each record in the emp table by passing on the salary value from the current record. The procedure displays the corresponding employee number, employee name and the salary grade returned by FIND_SAL_GRADE, on the screen. The procedure should handle error thrown by the function by displaying an appropriate message.
- Give a call to CALL_FIND_SAL_GRADE through an anonymous PL/SQL block

PACKAGES

Practice 7

- Create a package named MANAGE_EMP_PACK that has two public procedures, two package level variables and a private function. The public procedure HIRE_EMP adds an employee record in EMP table and the public procedure FIRE_EMP deletes an employee record from the EMP table. The two variables v_insert_cnt and v_delete_cnt are used in the package, for keeping record of the numbers of times insert / delete has been executed.

Create a private function VALIDATE_EMP in the package to validate employee number. This function can be called from HIRE_EMP and FIRE_EMP.

- The function VALIDATE_EMP accepts an employee number in a parameter p_eno and returns TRUE if the value of employee number exists in the EMP table else it returns FALSE.
- The procedure HIRE_EMP takes all the column values of the EMP table as parameters. It gives a call to VALIDATE_EMP by passing on the value of employee number and if the function returns TRUE then it displays message 'Employee number already in use'. If the function returns FALSE then it inserts a new record in the EMP table and displays a message 'One employee added'. It also increments the value of v_insert_cnt by 1.
- The procedure FIRE_EMP accepts an employee number as a parameter and gives a call to VALIDATE_EMP by passing on the value of employee number. If the function returns TRUE then it deletes the corresponding record from the EMP table, displays message 'One employee deleted' and increments the value of v_delete_cnt by 1. If the function returns FALSE then it displays message 'Wrong employee number'.
- Check working of the methods in MANAGE_EMP_PACK by giving calls to the public procedures and by displaying value of the appropriate package variable, through an anonymous block

Practice 8

- Create a package named MY_EMP_PACK having two **overloaded functions** named GET_AVG_SAL. The first function accepts ename as a parameter while the second function accepts empno as a parameter and both return the average salary paid in the department to which the employee belongs. Both the functions should handle the exception for non-existing employee by displaying an appropriate error message.
- Give a call to GET_AVG_SAL through an anonymous PL/SQL block by passing on employee number 7839. Do the same again by passing on employee name 'KING'.