**NAME : MANGESH A. GHADWAJE**

**ROLL NO : 24**

**BATCH : B2**

**COURSE: DBMS PRACTICAL**

**Assignment No :14**

**Problem Statement:-**

**1) Write a package that includes a procedure to validate user input for a specific data type and handle any exceptions that may occur.**

**2) Write a PL/SQL package that contains a function to retrieve the top N employees with the highest salaries. Table: employees( employee\_id integer first name varchar(25) last\_name varchar(25) email archar(25) phone\_number varchar(15) hire\_date date job\_id varchar(25) salary integer commission\_pct decimal(5,2) manager\_id integer department id integer)**

**Solution :**

### 1. Package to Validate User Input and Handle Exceptions

#### Package Specification

**CREATE OR REPLACE PACKAGE input\_validator\_pkg AS**

**PROCEDURE validate\_integer(input\_value VARCHAR2);**

**END input\_validator\_pkg;**

**/**

#### Package Body

**CREATE OR REPLACE PACKAGE BODY input\_validator\_pkg AS**

**PROCEDURE validate\_integer(input\_value VARCHAR2) IS**

**input\_as\_integer INTEGER;**

**BEGIN**

**-- Attempt to convert the input value to an integer**

**input\_as\_integer := TO\_NUMBER(input\_value);**

**-- If conversion is successful, print a message**

**DBMS\_OUTPUT.PUT\_LINE('Input value "' || input\_value || '" is a valid integer: ' || input\_as\_integer);**

**EXCEPTION**

**WHEN VALUE\_ERROR THEN**

**-- Handle the exception if input value is not a valid integer**

**DBMS\_OUTPUT.PUT\_LINE('Error: Input value "' || input\_value || '" is not a valid integer.');**

**WHEN OTHERS THEN**

**-- Handle any other exceptions**

**DBMS\_OUTPUT.PUT\_LINE('Error: An unexpected error occurred.');**

**END validate\_integer;**

**END input\_validator\_pkg;**

**/**

### 2. Package to Retrieve Top N Employees with Highest Salaries

#### Package Specification

**CREATE OR REPLACE PACKAGE employee\_pkg AS**

**TYPE employee\_ref\_cursor IS REF CURSOR;**

**FUNCTION get\_top\_n\_employees(n IN INTEGER) RETURN employee\_ref\_cursor;**

**END employee\_pkg;**

**/**

#### Package Body

**CREATE OR REPLACE PACKAGE BODY employee\_pkg AS**

**FUNCTION get\_top\_n\_employees(n IN INTEGER) RETURN employee\_ref\_cursor IS**

**emp\_cursor employee\_ref\_cursor;**

**BEGIN**

**OPEN emp\_cursor FOR**

**SELECT employee\_id, first\_name, last\_name, email, phone\_number, hire\_date,**

**job\_id, salary, commission\_pct, manager\_id, department\_id**

**FROM (SELECT \***

**FROM employees**

**ORDER BY salary DESC)**

**WHERE ROWNUM <= n;**

**RETURN emp\_cursor;**

**END get\_top\_n\_employees;**

**END employee\_pkg;**

**/**

#### Validating User Input

**BEGIN**

**input\_validator\_pkg.validate\_integer('123'); -- Valid integer**

**input\_validator\_pkg.validate\_integer('ABC'); -- Invalid integer**

**END;**

**/**

#### Retrieving Top N Employees

**DECLARE**

**emp\_cursor employee\_pkg.employee\_ref\_cursor;**

**emp\_record employees%ROWTYPE;**

**BEGIN**

**-- Get top 5 employees with the highest salaries**

**emp\_cursor := employee\_pkg.get\_top\_n\_employees(5);**

**-- Loop through the cursor and print employee details**

**LOOP**

**FETCH emp\_cursor INTO emp\_record;**

**EXIT WHEN emp\_cursor%NOTFOUND;**

**DBMS\_OUTPUT.PUT\_LINE('Employee ID: ' || emp\_record.employee\_id || ', Name: ' || emp\_record.first\_name || ' ' || emp\_record.last\_name || ', Salary: ' || emp\_record.salary);**

**END LOOP;**

**-- Close the cursor**

**CLOSE emp\_cursor;**

**END;**

**/**