Oracle MOOC: Introduction to PL/SQL Program Units

Week 2

Homework for Lesson 2: Handling Exceptions in PL/SQL

Homework is your chance to put what you've learned in this lesson into practice. This homework is not "graded" and you are encouraged to write additional code beyond what is asked.

Note:

- Ensure you complete the setup instructions provided on the course page before attempting the homework.
- The solutions to the homework are NOT provided. We encourage you to take the homework as a challenge and provide your own solutions. You can also use the course forum to collaborate on the solution with your fellow students.
- The homework is NOT mandatory to get the course completion award.
- Post your questions, comments, or suggestions (if any) in the course forum @ https://community.oracle.com/community/technology_network_community/moocs/plsql-program-units

Watch out for:

- Reference video that discussed the corresponding concept in this MOOC.

- Hints that can help you solve the assignment.

Assignment 1: In this practice, you write a PL/SQL block that applies a predefined exception to process only one record at a time. This block selects the name of the faculty with a given salary value and inserts the details into the `MESSAGES` table. In case, the select query returns multiple rows or no rows, it inserts an appropriate message into the `MESSAGES` table.

- Execute the command in the `lab_02_01.sql` file to create the `MESSAGES` table.

Oracle MOOC: PL/SQL Fundamentals

- In the declarative section, use the `%TYPE` attribute to declare two variables `v_fname` and `v_fac_sal` to hold the `last_name` and `salary` values from the `ad_faculty_details` table.
- Initialize the `v_fac_sal` to 6000.
- In the executable section, retrieve the last names of faculties whose salaries are equal to the value in `v_fac_sal`. **Note:** Do not use explicit cursors.
    - If the salary entered returns only one row, insert the faculty's name and the salary amount into the `MESSAGES` table. For example, `Louis - 6000`.
    - If the salary entered does not return any rows, handle the exception with an appropriate exception handler and insert "`No faculty with a salary of <salary>`." into the `MESSAGES` table.
    - If the salary entered returns multiple rows, handle the exception with an appropriate exception handler and insert "`More than one faculty with a salary of <salary>`." into the `MESSAGES` table.
    - Handle any other exception with an appropriate exception handler and insert the message "`Some other error occurred`." into the `MESSAGES` table.
- After executing this PL/SQL block, display the rows from the `MESSAGES` table to check whether the PL/SQL block has executed successfully.

Sample output:

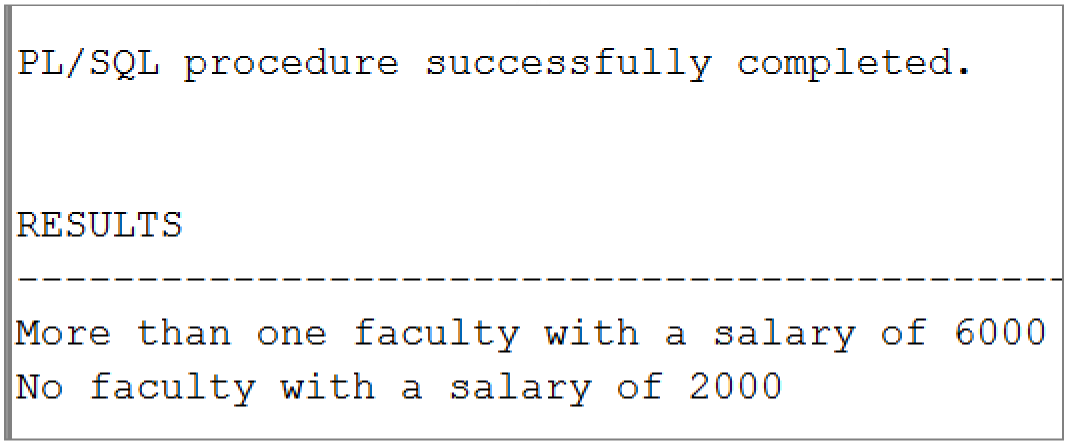
```
PL/SQL procedure successfully completed.


RESULTS
---------------------------------------------
More than one faculty with a salary of 6000
```

Oracle MOOC: PL/SQL Fundamentals

- Change the initialized value of `v_fac_sal` to 2000 and re-execute. Display the rows from the `MESSAGES` table to check whether the PL/SQL block has executed successfully.

Sample output:

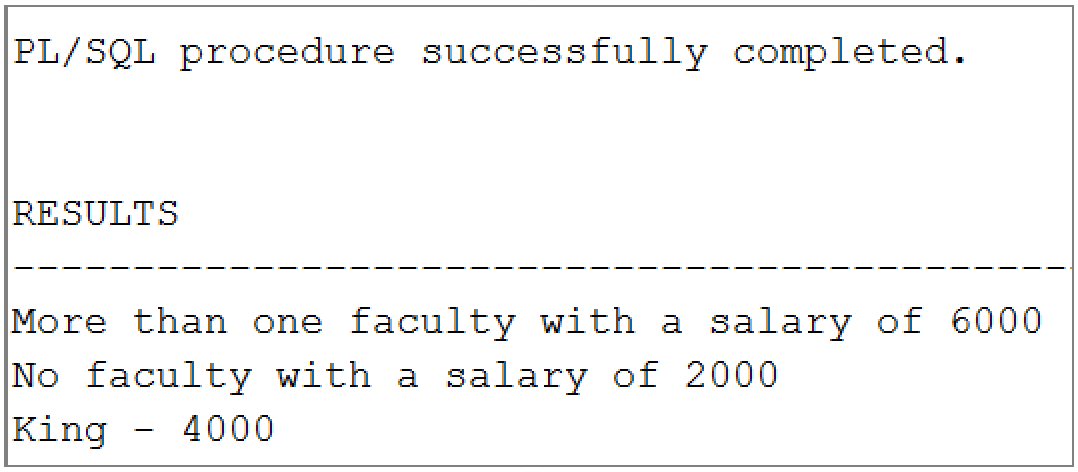


```
PL/SQL procedure successfully completed.


RESULTS
---------------------------------------------
More than one faculty with a salary of 6000
No faculty with a salary of 2000
```

- Change the initialized value of `v_fac_sal` to 4000 and re-execute. Display the rows from the `MESSAGES` table to check whether the PL/SQL block has executed successfully.

Sample output:



```
PL/SQL procedure successfully completed.


RESULTS
---------------------------------------------
More than one faculty with a salary of 6000
No faculty with a salary of 2000
King - 4000
```

Oracle MOOC: PL/SQL Fundamentals

See 2-2: Handling Predefined Exceptions for reference.

Oracle MOOC: PL/SQL Fundamentals

Assignment 2: In this practice, you write a PL/SQL block that declares an exception for the internally defined exception `ORA-02292` (integrity constraint violated – child record found). The block tests for the exception and outputs the error message.

- In the declarative section, declare an exception `e_childrecord_exists`. Associate the declared exception with the standard Oracle Server error `-02292`.
- In the executable section, display "`Deleting department 40....`" followed by a `DELETE` statement to delete the department with `department_id` 40 from `ad_departments` table.
- Include an exception section to handle the `e_childrecord_exists` exception and display the appropriate message.

Sample output:

```
 Deleting department 40........
 Cannot delete this department.
    There are
active courses in this department (child records exist.)


PL/SQL procedure successfully completed.
```

See 2-3: Handling Internally Defined Exceptions for reference.

Assignment 3: In this practice, you write a PL/SQL block that handles two user-defined exceptions while inserting a new row into the `ad_exam_results` table. This PL/SQL block accepts course id and marks as input values and inserts a row with these values for the `student_id` = 720 and `exam_id` = 520, into the `ad_exam_results` table. Before inserting the row, it performs the following validations and raises appropriate user-defined exceptions.

Oracle MOOC: PL/SQL Fundamentals

- Is the student with `student_id` = 720 enrolled into the input course id?
- Are the input marks within limits? That is, the marks are not greater than 100.

Hints:

- In the declarative section, declare two variables of type exception as `e_invalid_course` and `e_marks_out_of_range`.
- Declare variables `v_course_id` and `v_marks` using the `%TYPE` attribute.
- Initialize `v_course_id` and `v_marks` to `& b_course_id` and `& marks` respectively, to accept input values.
- Declare the following variables and initialize them as below:
    - `v_student_id` = 720
    - `v_exam_id` = 520
    - `v_count` = 0
- In the executable section, query the `ad_student_course_details` table to find out how many rows have `student_id` = 720 and `course_id` = input course id that is `v_course_id`. Store the retrieved value into `v_count`.
- If v`_count` = 0, then raise the user-defined exception `e_invalid_course`.
- If `v_marks` > 100, then raise the user-defined exception `e_marks_out_of_range`.
- If no exception is raised, insert a new row with `v_student_id`, `v_exam_id`, `v_course_id`, `v_marks` into the `ad_exam_results` table.
- Handle `e_invalid_course` exception in the exception block to display a message: “This student is not enrolled into <course id>.”
- Handle `e_marks_out_of_range` exception in the exception block to display a message: “The input marks are out of range”.
- Execute this PL/SQL block with input course id = 195 and marks = 95, and see the output.

Oracle MOOC: PL/SQL Fundamentals

Sample output:

```
This student is not enrolled into 195


PL/SQL procedure successfully completed.
```

- Execute this PL/SQL block with input course id = 190 and marks = 125, and see the output.

Sample output:

```
The input marks are out of range.


PL/SQL procedure successfully completed.
```

- Execute this PL/SQL block with input course id = 195 and marks = 125, and see the output.

Sample output:

```
This student is not enrolled into 195


PL/SQL procedure successfully completed.
```

See 2-4: Handling User-defined Exceptions for reference.

Congratulations! You successfully practiced the concepts discussed in week 2.