8-2

Agustín Alejandro Mota Hinojosa

December 3, 2023

Contents

1 Vocabulary

1. Parameter Passing: Pass or communicate data between the caller and subprogram.
2. Actual Parameter: The actual value assigned to a parameter.
3. Actual Argument: Can be literal values, variables, or expressions that are provided in the parameter list of a called subprogram.
4. Formal Parameter: A parameter name declared in the procedure heading.

2 Try

1. Parameters in PL/SQL subprograms are variables used to pass values into or out of subprograms such as procedures and functions.
   - There are two types of parameters:
     - IN Parameters: Used for passing values into the subprogram. The values are read-only within the subprogram.
     - OUT Parameters: Used for passing values out of the subprogram. The subprogram can modify the values of OUT parameters.
     - IN OUT Parameters: Used for both passing values into and out of the subprogram.
2. COUNTRIES table:
   (a) code:

```
CREATE OR REPLACE PROCEDURE get_country_info(
   p_country_id IN COUNTRIES.COUNTRY_ID%TYPE
) AS
   v_country_name COUNTRIES.COUNTRY_NAME%TYPE;
   v_capital COUNTRIES.CAPITAL%TYPE;
BEGIN
   SELECT COUNTRY_NAME, CAPITAL INTO v_country_name, v_capital
   FROM COUNTRIES
   WHERE COUNTRY_ID = p_country_id;

   DBMS_OUTPUT.PUT_LINE('Country: ' || v_country_name || ', Capital: ' || v_capital);
```

```
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('No information found for Country ID ' || p_country_id);
END get_country_info;
/
```

(b) execute:

```
BEGIN
   get_country_info(90);
END;
```

(c) re-execute: This would display the information for the country with ID 95 if it exists; otherwise, it will handle the $\text{NO}_{\text{DATAFOUND}}$ exception.

(d) Modify

```
CREATE OR REPLACE PROCEDURE get_country_info(
   p_country_id IN COUNTRIES.COUNTRY_ID%TYPE
) AS
   v_country_name COUNTRIES.COUNTRY_NAME%TYPE;
   v_capital COUNTRIES.CAPITAL%TYPE;
BEGIN
   BEGIN
      SELECT COUNTRY_NAME, CAPITAL INTO v_country_name, v_capital
      FROM COUNTRIES
      WHERE COUNTRY_ID = p_country_id;

      DBMS_OUTPUT.PUT_LINE('Country: ' || v_country_name || ', Capital: ' || v_capital);
   EXCEPTION
      WHEN NO_DATA_FOUND THEN
         DBMS_OUTPUT.PUT_LINE('No information found for Country ID ' || p_country_id);
   END;
END get_country_info;
/
```

3. Formal Parameter: Parameters declared in the subprogram specification or body. They define the interface of the subprogram.

4. Procedure for Region's Highest Elevation:

(a) code:

```
CREATE OR REPLACE PROCEDURE country_count_by_elevation(
   p_region_id IN REGIONS.REGION_ID%TYPE,
   p_min_elevation IN NUMBER,
   p_first_char IN CHAR DEFAULT NULL
) AS
   v_count NUMBER;
BEGIN
   SELECT COUNT(*)
   INTO v_count
   FROM COUNTRIES c
   JOIN LOCATIONS l ON c.LOCATION_ID = l.LOCATION_ID
   WHERE c.REGION_ID = p_region_id
      AND l.HIGHEST_ELEVATION > p_min_elevation
      AND (p_first_char IS NULL OR SUBSTR(c.COUNTRY_NAME, 1, 1) = p_first_char);
```

```
   DBMS_OUTPUT.PUT_LINE('Number of countries: ' || v_count);
END country_count_by_elevation;
/
```

(b) Execute:

```
BEGIN
   country_count_by_elevation(5, 2000);
END;
```

(c) DESC table: Can't because I'm not in apex

(d) modify code:

```
CREATE OR REPLACE PROCEDURE country_count_by_elevation(
   p_region_id IN REGIONS.REGION_ID%TYPE,
   p_min_elevation IN NUMBER,
   p_first_char IN CHAR DEFAULT NULL
) AS
   v_count NUMBER;
BEGIN
   SELECT COUNT(*)
   INTO v_count
   FROM COUNTRIES c
   JOIN LOCATIONS l ON c.LOCATION_ID = l.LOCATION_ID
   WHERE c.REGION_ID = p_region_id
      AND l.HIGHEST_ELEVATION > p_min_elevation
      AND (p_first_char IS NULL OR SUBSTR(c.COUNTRY_NAME, 1, 1) = p_first_char);

   DBMS_OUTPUT.PUT_LINE('Number of countries: ' || v_count);
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('No countries found for the specified criteria.');
END country_count_by_elevation;
/
```

(e) Anonymous Block

```
   DECLARE
      v_region_id REGIONS.REGION_ID%TYPE := 5;
      v_elevation NUMBER := 2000;
      v_first_char CHAR := 'B';
   BEGIN
      country_count_by_elevation(v_region_id, v_elevation, v_first_char);
   END;
```

(f) code

```
DECLARE
   v_region_id REGIONS.REGION_ID%TYPE := 5;
   v_elevation NUMBER := 2000;
   v_first_char CHAR := 'B';
BEGIN
   country_count_by_elevation(v_first_char, v_region_id, v_elevation);
END;
```