PL/SQL part 2 of 3 Notes adapted from Nebojsa October 2020

Very similar to part 1 of 3 – another set of examples

How to build a Block without Exception Handler and then how to transform it to a Stored Procedure with one IN parameter, ***Part II***

Problem: For a given EMPLOYEE last name, you need to find out his/her salary and provide a comment (note) about that amount.

* Less than $3,000 🡪 POOR
* Less than $6,000 🡪 FAIR
* Less than $10,000 🡪 GOOD
* Less or equal than $15,000 🡪 EXCELLENT
* More than $15,000 🡪 WOW

You also need to be prepared for a name that does NOT exist and for the case when more than one person holds the same name.

1) Our first Code Example is a Block without Exception Handler. It will work properly only if for a given LAST NAME, there is ONLY ONE person.

**DECLARE**

**v\_lname employees.last\_name%TYPE := 'HIGGINS' ;**

**v\_pay employees.salary%TYPE;**

**v\_note VARCHAR2(20) := 'FAIR';**

**BEGIN**

**SELECT salary INTO v\_pay**

**FROM employees**

**WHERE UPPER(last\_name) = v\_lname;**

**IF v\_pay < 3000 THEN**

**v\_note := 'POOR';**

**ELSIF v\_pay < 6000 THEN**

**v\_note := 'FAIR';**

**ELSIF v\_pay < 10000 THEN**

**v\_note := 'GOOD';**

**ELSIF v\_pay <= 15000 THEN**

**v\_note := 'EXCELLENT';**

**ELSE**

**v\_note := 'WOW';**

**END IF;**

**DBMS\_OUTPUT.PUT\_LINE('Employee ' || v\_lname || ' Monthly income of $' || v\_pay || ' which is ' || v\_note);**

**END;**

SQL> **@ex4;** -- name given is HIGGINS (a single person with that name)

**Employee HIGGINS Monthly income of $12000 which is EXCELLENT**

PL/SQL procedure successfully completed.

SQL> **@ex4;** -- name given is GRANT There is no GRANT in your employees

**ORA-06512: at line 6**

**01403. 00000 - "no data found"**

**Now, we will add employee with a name Grant** twice

SQL> desc employee

Name Null? Type

----------------------------------------- -------- ----------------

EMPLOYEE\_ID NOT NULL NUMBER(6)

FIRST\_NAME VARCHAR2(20)

LAST\_NAME NOT NULL VARCHAR2(25)

EMAIL NOT NULL VARCHAR2(25)

PHONE\_NUMBER VARCHAR2(20)

HIRE\_DATE NOT NULL DATE

JOB\_ID NOT NULL VARCHAR2(10)

SALARY NUMBER(8,2)

COMMISSION\_PCT NUMBER(2,2)

MANAGER\_ID NUMBER(6)

DEPARTMENT\_ID NUMBER(4)

SQL> INSERT INTO employees (employee\_id, last\_name, email, hire\_date, job\_id,salary)

VALUES (901, 'Grant','pgrant@yahoo.ca',sysdate,'IT\_PROG',14000);

1 row created.

**INSERT AGAIN**

INSERT INTO employees (employee\_id, last\_name, email, hire\_date, job\_id,salary)

VALUES (902, 'Grant','grant2@yahoo.ca',sysdate,'IT\_PROG',14000);

Check to see it is done

**SELECT employee\_id, last\_name,job\_id,salary**

**FROM employees WHERE last\_name = 'Grant';**

EMPLOYEE\_ID LAST\_NAME JOB\_ID SALARY

----------- ------------------------- ---------- ----------

901 Grant IT\_PROG 14000

902 Grant IT\_PROG 14000

Run again with GRANT -- name given is GRANT (two persons with that name)

**ORA-01422: exact fetch returns more than requested number of rows**

**2)** Our second Code Example is a Block with Exception Handler that deals with BOTH exceptions, so you will not get Error messages

**DECLARE**

**v\_lname employees.last\_name%TYPE := 'ADAMS' ;**

**v\_pay employees.salary%TYPE;**

**v\_note VARCHAR2(20) := 'FAIR';**

**BEGIN**

**SELECT salary INTO v\_pay**

**FROM employees**

**WHERE UPPER(last\_name) = v\_lname;**

**IF v\_pay < 3000 THEN**

**v\_note := 'POOR';**

**ELSIF v\_pay < 6000 THEN**

**v\_note := 'FAIR';**

**ELSIF v\_pay < 10000 THEN**

**v\_note := 'GOOD';**

**ELSIF v\_pay <= 15000 THEN**

**v\_note := 'EXCELLENT';**

**ELSE**

**v\_note := 'WOW';**

**END IF;**

**DBMS\_OUTPUT.PUT\_LINE('Employee ' || v\_lname || ' has a monthly income of $' || v\_pay || ' which is ' || v\_note);**

**EXCEPTION**

**WHEN NO\_DATA\_FOUND THEN**

**DBMS\_OUTPUT.PUT\_LINE('Employee ' || v\_lname || ' does NOT exist');**

**WHEN TOO\_MANY\_ROWS THEN**

**DBMS\_OUTPUT.PUT\_LINE('There is more than one employee with such last name: ' || v\_lname);**

**END;**

Test with HIGGINS and GRANT to see they still work then

Test with ADAMS

**Employee ADAMS does NOT exist**

PL/SQL procedure successfully completed.

3) Our third Code Example is a Stored Procedure that accepts one IN parameter, LAST\_NAME. Watch how the code has changed, when using p\_lname and not v\_lname.

**CREATE OR REPLACE PROCEDURE find\_sal(p\_lname IN employees.last\_name%TYPE)**

**IS**

**v\_pay employees.salary%TYPE;**

**v\_note VARCHAR2(20) := 'FAIR';**

**BEGIN**

**SELECT salary INTO v\_pay**

**FROM employees**

**WHERE UPPER(last\_name) = p\_lname;**

**IF v\_pay < 3000 THEN**

**v\_note := 'POOR';**

**ELSIF v\_pay < 6000 THEN**

**v\_note := 'FAIR';**

**ELSIF v\_pay < 10000 THEN**

**v\_note := 'GOOD';**

**ELSIF v\_pay <= 15000 THEN**

**v\_note := 'EXCELLENT';**

**ELSE**

**v\_note := 'WOW';**

**END IF;**

**DBMS\_OUTPUT.PUT\_LINE('Employee ' || p\_lname || ' has a monthly income of $' || v\_pay || ' which is ' || v\_note);**

**EXCEPTION**

**WHEN NO\_DATA\_FOUND THEN**

**DBMS\_OUTPUT.PUT\_LINE('Employee ' || p\_lname || ' does NOT exist');**

**WHEN TOO\_MANY\_ROWS THEN**

**DBMS\_OUTPUT.PUT\_LINE('There is more than one employee with such last name: ' || p\_lname);**

**END;**

SQL> **EXECUTE find\_sal('WHALEN');**

**Employee WHALEN has a monthly income of $4400 which is FAIR**

PL/SQL procedure successfully completed.

SQL> **EXECUTE find\_sal('GRANT');**

**There is more than one employee with such last name: GRANT**

PL/SQL procedure successfully completed.

SQL> **EXECUTE find\_sal('DE NIRO');**

**Employee DE NIRO does NOT exist**

PL/SQL procedure successfully completed.

**Because we added two employees called GRANT we need to remove them**

DELETE FROM employees

WHERE employee\_id IN (901,902);

1 row deleted.

SQL> commit;

Commit complete.

SQL> SELECT employee\_id, last\_name,job\_id,salary

FROM employees

WHERE last\_name = 'Grant'

No rows seleted