1) Write the PL/SQL block that will add a new row to the Departments table like shown:

**Department\_id** 🡪 Highest value of all Id’s increased by 50

**Department\_name** 🡪 Testing

**Manager\_id** 🡪 One with most people under supervision (do NOT assume it is a President, prove it with the code)

**Location\_id** 🡪 You will be asked to input the City name where you want to place your new department. And that value will be converted to its related Location\_id.

You will then display that new row and later undo the insert.

Your input value should be a VALID city name without any department. You need also to code for following cases: There is already ONE department in that city, there is MORE THAN ONE department there and the city is NOT listed in the Locations table.

**You do NOT need to use cursor for this question,** Here are the possible outputs:

Top of Form

|  |  |  |
| --- | --- | --- |
| Please provide the valid city without department: | t |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **DEPARTMENT\_ID** | **DEPARTMENT\_NAME** | **MANAGER\_ID** | **LOCATION\_ID** |
| 320 | Testing | 100 | 1100 |

**Rollback complete.**

Top of Form

|  |  |  |
| --- | --- | --- |
| Please provide the valid city without department: | t |  |

**This city already contains department: Toronto  
PL/SQL procedure successfully completed.**

**no rows selected   
Rollback complete.**

Top of Form

|  |  |  |
| --- | --- | --- |
| Please provide the valid city without department: | t |  |

**This city has MORE THAN ONE department: Seattle  
PL/SQL procedure successfully completed.  
  
no rows selected   
Rollback complete.**

Top of Form

|  |  |  |
| --- | --- | --- |
| Please provide the valid city without department: | t |  |

**This city is NOT listed: Belgrade  
PL/SQL procedure successfully completed.**

**no rows selected   
Rollback complete.**

VARIABLE v\_newcityid NUMBER

SET SERVEROUTPUT ON

ACCEPT city PROMPT 'Please Enter a valid city without department : '

SET VERIFY OFF

DECLARE

v\_city locations.city%TYPE := '&city';

loc# locations.location\_id%TYPE;

dept\_count INTEGER := 0 ;

emp\_count INTEGER := 0 ;

v\_dpt\_id departments.department\_id%TYPE;

v\_mgr\_id employees.manager\_id%TYPE;

BEGIN

-- first getting the location id of the entered city

SELECT location\_id INTO loc#

FROM locations

WHERE UPPER(city) IN UPPER(v\_city);

--Getting department count under the city

SELECT COUNT(department\_id) INTO dept\_count

FROM

departments

WHERE location\_id = loc#;

--assigning dept id

SELECT MAX(department\_id)+50 INTO v\_dpt\_id FROM departments;

--assigning manager\_id

SELECT count(e.employee\_id),m.manager\_id INTO emp\_count, v\_mgr\_id

FROM employees e, employees m WHERE e.manager\_id = m.employee\_id

HAVING COUNT(e.employee\_id) =

(SELECT MAX(COUNT(e.employee\_id)) FROM employees e, employees m

WHERE e.manager\_id = m.employee\_id

GROUP BY m.manager\_id)

GROUP BY m.manager\_id;

-- Putting conditions on that number of departments in that city

CASE

WHEN dept\_count = 0 THEN

INSERT INTO departments VALUES(v\_dpt\_id, 'Testing' , v\_mgr\_id, loc#);

SELECT location\_id INTO :v\_newcityid from locations WHERE city=v\_city;

WHEN dept\_count = 1 THEN

DBMS\_OUTPUT.PUT\_LINE('This city already contains department: ' || v\_city);

WHEN dept\_count > 1 THEN

DBMS\_OUTPUT.PUT\_LINE('This city has MORE THAN ONE department: ' || v\_city);

END CASE;

EXCEPTION

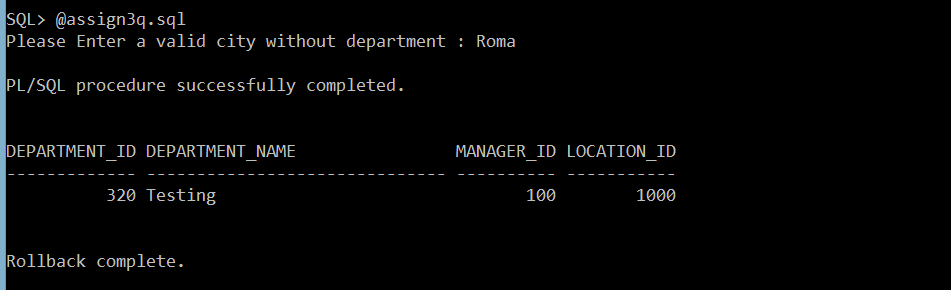
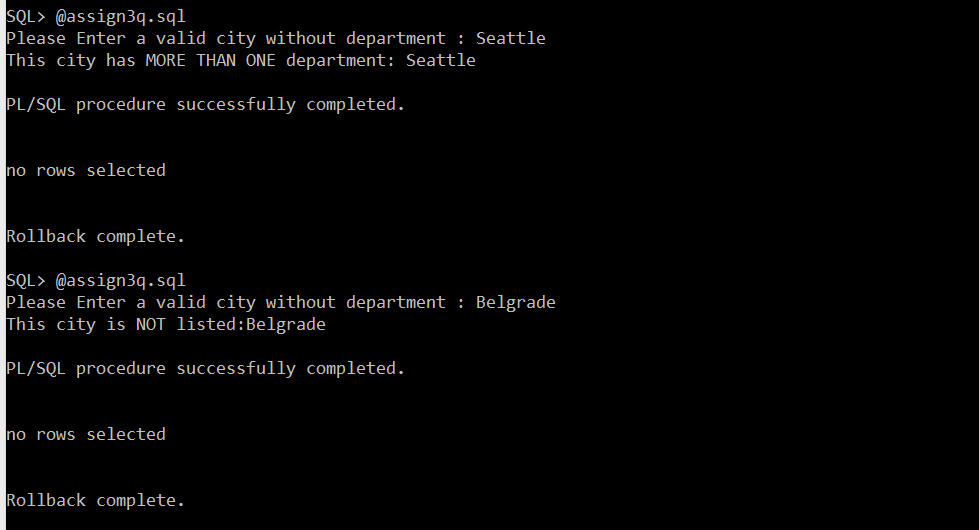
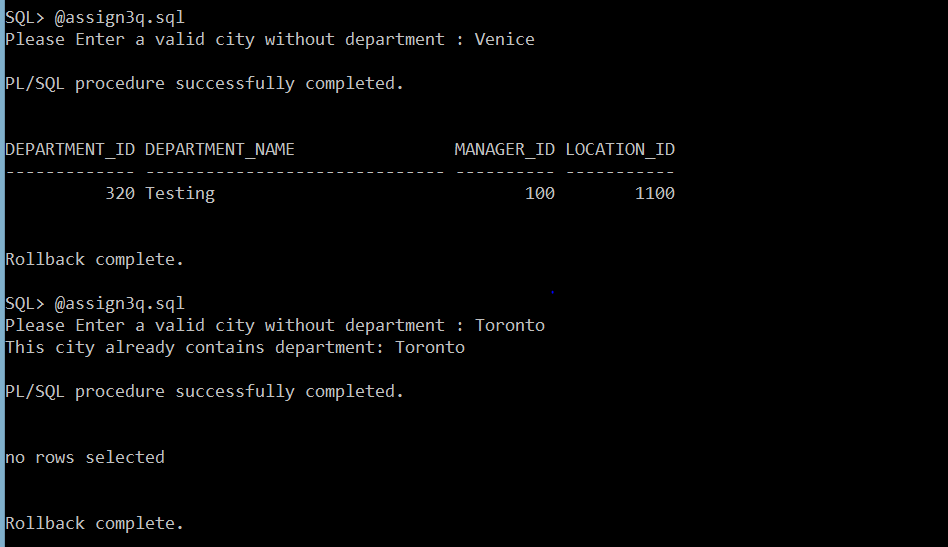
WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('This city is NOT listed:' || v\_city);

END;

/

SELECT \* FROM departments WHERE location\_id=:v\_newcityid;

ROLLBACK; 

2) Write the PL/SQL block that will as input accept the start of the Course Description (one or more words) that possibly has a PREREQUISITE course. Then, it will display for each course that starts like the provided input the following components: Course Number, Description and Cost followed by the prerequisite Course Number, Description **and Cost**.. **Use For Cursor Loop.**

Also, you need to deal with the situations where your input points to a valid course that has NO prerequisites at all and also if your input points to a non-existing course. Here are the possible outputs:

Top of Form

|  |  |  |
| --- | --- | --- |
| Enter the beginning of the Course description in UPPER case: | t |  |

**Course: 144 - Database Design   
Cost: 1195   
Prerequisite: 420 - Database System Principles   
Prerequisite Cost: 1195   
========================================   
Course: 420 - Database System Principles   
Cost: 1195   
Prerequisite: 25 - Intro to Programming   
Prerequisite Cost: 1195   
========================================   
PL/SQL procedure successfully completed.**

Top of Form

|  |  |  |
| --- | --- | --- |
| Enter the beginning of the Course description in UPPER case: | t |  |

**There is NO prerequisite course for any course that starts on OPERATING. Try again.   
PL/SQL procedure successfully completed.**

Top of Form

|  |  |  |
| --- | --- | --- |
| Enter the beginning of the Course description in UPPER case: | t |  |

**There is NO VALID course that starts on SPORT. Try again.   
PL/SQL procedure successfully completed.**

**Answer 2:**

SET SERVEROUTPUT ON

ACCEPT subject PROMPT 'Enter the beginning of the Course description in UPPER case: '

SET VERIFY OFF

DECLARE

v\_course\_desc VARCHAR(20):='&subject'; -- keeping track of subject user entered

-- storing prerequisite details in the following variables

v\_prerequisite course.prerequisite%TYPE;

v\_predesc course.description%TYPE;

v\_precost course.cost%TYPE;

-- cursor to get all details of subject details user entered

CURSOR cursor\_subject IS

SELECT \* FROM course WHERE UPPER(description) LIKE UPPER('&subject%');

subject\_details cursor\_subject%ROWTYPE;

BEGIN

-- first checking for the presence of any subject with user subject details entered

OPEN cursor\_subject;

FETCH cursor\_subject INTO subject\_details;

-- if the subject details entered by the user is not found then

IF cursor\_subject%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('There is NO VALID course that starts on '||v\_course\_desc||'. Try again. ');

CLOSE cursor\_subject;

-- if the subject details entered by the user is found then

ELSE

CLOSE cursor\_subject;

FOR i IN cursor\_subject LOOP

-- checking if the subject entered has a prerequisite

IF i.prerequisite IS NOT NULL THEN

DBMS\_OUTPUT.PUT\_LINE('Course: '||i.course\_no||' - '||i.description);

DBMS\_OUTPUT.PUT\_LINE('Cost: '||i.cost);

-- if the subject has a prerequisite then get its details and print them

SELECT p.course\_no,p.description,p.cost INTO v\_prerequisite,v\_predesc,v\_precost

FROM course c RIGHT OUTER JOIN course p

ON c.prerequisite=p.course\_no

WHERE c.course\_no=i.course\_no;

DBMS\_OUTPUT.PUT\_LINE('Prerequisite: '||v\_prerequisite||' - '||v\_predesc);

DBMS\_OUTPUT.PUT\_LINE('Cost: '||v\_precost);

DBMS\_OUTPUT.PUT\_LINE('==============================================');

-- if no prerequisite found then display appropriate msg

ELSE

DBMS\_OUTPUT.PUT\_LINE('There is NO prerequisite course for any course that starts on '||v\_course\_desc||'. Try again.');

END IF; -- finishing of inner if loop

END LOOP;

END IF; -- finishing of outer if loop

END;

/

