Home assignment #5 (352 &452),

Due date 5/18/2022

Q1. (20 points).

A very important part of DB is about the system catalog, in hw 3 question 3, we have introduced the view All\_Tab\_Columns.

In this question will use a view from system catalog called user\_constraints. (Similarly, there is another view named user\_cons\_columns).You may use ‘desc user\_constraints’ to see the columns defined in this table/view. You may refer to this link to learn more.

<https://docs.oracle.com/cd/E18283_01/server.112/e17110/statviews_1046.htm>

You need to define a user-defined record type called Constrnt, which has three columns:

T\_name varchar2 (30) -- correspondent to table\_name

const\_name varchar2 (30) -- correspondent to constraint\_name

Const\_type char (1) -- correspondent to constraint\_type

Write an anonymous PL/SQL block, define a cursor for retrieving the info from the view of

user\_constraints for the table named EMPLOYEES and Departments. In the executable section, your program will run a loop to fetch the cursor (values) into a record variable just defined, then print out the contents of this record. Better to use basic loop, as it is easier to “INTO” the local variable.

The structure below is for your reference,

Cursor c

IS

select table\_name, constraint\_name , constraint\_type

from user\_constraints

where upper (table\_name) in ( 'EMPLOYEES' , 'DEPARTMENTS')

order by 1;

Description of constraint type LETTER:

P - Primary key.

U - Unique key.

R - Referential integrity.

C - Check constraint on a table.

Q2. (20 points). Refer to page 3 in note6b\_Records.

Define a user-defined record type called DEPT\_INFO that has three fields as:

dept\_id (correspondent to department\_id in departments table),

dept\_name (department\_name in departments table),

total\_emp as integer, it should be the number ( how many) employees work in this department\_id, the info should be retrieved from employees table.

After having defined this record type, then your program will declare a variable of this type.

The program will retrieve the needed info for department\_id 60, populate each field of this record type variable, and print out the result.

Note: we can either use a single SQL to retrieve all correspondent columns for the fields in this records; or we may use a few separated SQL statements just coping with one field value at a time. For example, as for department\_id 60:

SELECT count (\*) as total\_emp FROM employees WHERE department\_id = 60;

There are some hints provided in the end of this file for in case you need.

Q3. (20 points)

In question 2, we build a record type, and declare a variable of that type, it is good to store one record info at a time. For this question, we will create a collection using this record type as its elements, thus it can store a set records.

Write a PL/SQL block, define a user-defined record type called Dept\_Info as in question 2 that has three fields.

Define a nested table type named as Dept\_Info\_NT, that uses Dept\_Info as its element; then declare a nested table variable called LIST.

Retrieve data from Departments and Employees table for all the departments that have a manager for that department (departments.manager\_id is not null) assigned. Populate this LIST with the retrieved data, print out the contents of the LIST.

Note, we use NT\_variable\_name (index) to reference the NT element; in this question, we need to use NT\_variable\_name (index).record\_field to reference the element’s field. Such as

LIST(i).dept\_id for department\_id column in the departments table.

If you have difficulties, please refer to the hints at end of this file.

Q4. (15 points) note, Q5 and Q6 are based on this question Q4.

Write a PL/SQL block, in that program, define a local procedure named Dept\_Head\_Name. That means this procedure lives in that block only, it is also called nested procedure.

This procedure will accept the Department\_ID as IN parameter, and it has one OUT parameter, called Dept\_Head\_fullname that will pass back the department manager’s full name (first name + space + last name) to the invoker (program).

In the executable section, the program will invoke this procedure Dept\_Head\_Name with the department ID of 60 as IN parameter value. The program will print out the full name of the manager of that department.

There is some hints for the SQL if needed.

Q5. (10 points)

This question 5 is based on question 4. This time, you will create a standalone procedure same as in question 4, all you will do is to change the local procedure into a standalone procedure. Your program will consist of two parts. The first part is a block of “create or replace procedure” .You need to compile this “create procedure ” code before you invoke that procedure.

After having successfully compiled the stand alone procedure, as second part, you will write a separate PL/SQL block, in its executable section, the program will invoke this procedure passing the value of 60 of department ID to the procedure, your program will store the OUT parameter value into your local variable. Your program will then print out the manager full name of this department 60.

Simply say, change the code in q4, make the procedure as a stand alone via create or replace it in the DB schema level. Then you may call it in your PL/SQL block as in Q4.

Q6. (15 points) Based on question 5.

This time, you will print out the full names of the department’s head ( manager) for all the departments that has a manager assigned to.

Assuming that you have created the procedure in Q5, so you do not need to repeat your “create procedure ” code, just write one PL/SQL that will invoke standalone procedure.

In the PL/SQL block, you will define a cursor that will return all the department ID’s

(... from departments where manager\_id is not null),

run a loop, repeatedly invoke that procedure. Your program needs to provide the value of department\_id you get from cursor to the procedure, and assign the OUT parameter value from the procedure into your local variable (the full name). Then print out the info.

**Hints.**

Q1**.**

The is for your reference,

Cursor c

IS

select table\_name, constraint\_name , constraint\_type

from user\_constraints

where upper (table\_name) in ( 'EMPLOYEES' , 'DEPARTMENTS')

order by 1;

Q2.

you may use one SQL statement:

select e.department\_id , d.department\_name, count (e.employee\_id) total

from employees e, departments d

where e.department\_id = d.department\_id and e.department\_id = 60

group by e.department\_id , d.department\_name;

Q3.

this SQL statement may help to define a cursor:

select e.department\_id , d.department\_name dname, count (e.employee\_id) total

from employees e, departments d

where e.department\_id = d.department\_id and d.manager\_id is not null

group by e.department\_id, d.department\_name

order by 1;

Do not forget to allocate the storage with calling the List.extend procedure, such as:

For indx in c loop -- cursor for loop

LIST.extend;

LIST(i).dept\_id := indx.department\_id;

...

Q4. Hint. When use this query in PL/SQL block, you need to change the literal value 60 into a variable name.

SELECT first\_name ||' '|| last\_name

FROM employees WHERE employee\_id =

(select manager\_id from departments where department\_id = 60);

-- or

SELECT e.department\_id, first\_name ||' '|| last\_name as dept\_head\_name

FROM employees e, departments d

WHERE e.employee\_id = d.manager\_id and e.department\_id = 60 ;