Vishnu Remesh

11327223

ASSIGNMENTS

1) Accept job type from the user. Display the message depending upon whether

no rows or one row or several rows are selected.

The message should be any one from the below 3 as per the situation.

JOB TYPE FOUND ONCE

JOB TYPE FOUND MORE THAN ONCE

JOB TYPE NOT FOUND

ANS: declare

v\_emp VARCHAR2(10);

begin

select ename

into v\_emp

from emp

where job='&a';

dbms\_output.put\_line('JOB TYPE FOUND ONCE');

exception

when no\_data\_found then

dbms\_output.put\_line('JOB TYPE NOT FOUND');

WHEN too\_many\_rows then

dbms\_output.put\_line('JOB TYPE MORE THAN ONCE');

end;

2) Using basic loop technique display all the multiples of 7 between 31 and 48.

ANS: declare

num1 number(10):=28;

begin

loop

exit when num1>48;

num1:=num1+7;

dbms\_output.put\_line(num1);

end loop;

end;

3) Write a PL/SQL code to display the name, salary and grade of the employee by accepting employee code. Grade is ‘A’ if salary >25000, ‘B’ if salary > 15000, ‘C’ for other salaries. Use CASE statement only.

ANS: create or replace function USER\_VALID\_DEPTNO(p\_dno in NUMBER) RETURN varchar2

is

begin

if(p\_=10 OR p\_no=20 or p\_dno=30 or p\_dno=40) THEN

return 'true';

else

return 'false';

end if;

end;

create or replace procedure SHOW\_STRENGTH(p\_deptno in NUMBER)

is

v1 varchar(10);

nam number(10);

begin

v1:=USER\_VALID\_DEPTNO(p\_deptno);

if v1='true' then

select count(ename)

into nam

from emp

where deptno= p\_deptno;

dbms\_output.put\_line(to\_char(nam));

else

dbms\_output.put\_line('NO DATA');

end if;

exception

when no\_data\_found THEN

dbms\_output.put\_line('NO DATA');

end;

begin

SHOW\_STRENGTH(10);

end;

4) Check whether the 3rd Clerk’s salary is greater than the 2nd Clerk’s salary. If it is so, then display message as “Third has more salary than the second” otherwise display message as “Second has more salary than the first”

ANS: declare

cursor c1 is

select sal

from emp

where job='CLERK';

temp3 c1%rowtype;

temp2 c1%rowtype;

emprec c1%rowtype;

begin

open c1;

fetch c1 into emprec;

fetch c1 into temp2;

fetch c1 into temp3;

if(temp3.sal>temp2.sal) THEN

dbms\_output.put\_line('Third has more salary than the second');

else

dbms\_output.put\_line('Second has more salary than the first');

end if;

close c1;

end;

5) The management has decided to increase salary of employees by 10%. Starting with lowest paid earner and working up. If at any time salary bill exceeds 45000/- then no further employees are to be given an increase.

ANS: declare

cursor c1 is

select sal from emp

order by sal;

temp1 c1%rowtype;

emprec c1%rowtype;

temp2 NUMBER(10):=0;

begin

open c1;

loop

fetch c1 into emprec;

temp1.sal:=emprec.sal\*0.1+emprec.sal;

temp2:=temp2+temp1.sal;

exit when temp2>45000;

dbms\_output.put\_line(temp2);

end loop;

close c1;

end;

6) Use a parameterized cursor which will take deptno as the parameter and will display the highest two salaries of that deptno.

ANS: declare

cursor c1(vnum number) is

select sal from emp

where deptno=vnum

order by sal desc;

emprec c1%rowtype;

begin

open c1(10);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

close c1;

open c1(20);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

close c1;

open c1(30);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

close c1;

open c1(40);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

fetch c1 into emprec;

dbms\_output.put\_line(emprec.sal);

close c1;

end;

7) ) Create the table Emp\_Coupons with the two fields Name and Coupon\_No

Enter the following records in it.

John 80

Martin 83

Allen 87

Roger 78

Adams 88

Kim 89

Make a PLSQL block that will check whether the coupon number of the current record is greater than the previous. If any record’s coupon number is less than the previous one then display the name of the person whose coupon number is less.

ANS: declare

cursor c1 is

SELECT coupons from Emp\_Coupons;

temp1 c1%rowtype;

temp2 c1%rowtype;

begin

open c1;

loop

fetch c1 into temp1;

loop

fetch c1 into temp2;

if temp1.coupons>temp2.coupons then

dbms\_output.put\_line('First greater than second');

else

dbms\_output.put\_line('second greater than first');

end if;

exit when c1%notfound;

end loop;

exit when c1%notfound;

end loop;

close c1;

end;

8) PLSQL Table Assignments

Add multiples of 5 from 5 to 130 in a plsql table and display them from the plsql table in the descending order.

ANS: declare

type nametable is table of number(10) index by binary\_integer;

num number(10):=5;

d nametable;

n number(10):=1;

begin

loop

exit when num>130;

d(n):=num;

n:=n+1;

num:=num+5;

end loop;

for i in reverse 1..26 loop

dbms\_output.put\_line(d(i));

end loop;

end;

9) Accept the name of the employee. Display the salary of that employee.

Handle all possible run-time errors.

ANS: create or replace procedure p\_sal(p\_name in VARCHAR2,p\_sal out number) is

begin

select sal into p\_sal

from emp

where ename=p\_name;

exception when no\_data\_found then

dbms\_output.put\_line('No employee with such name');

end;

declare

v1 number(10);

begin

p\_sal('&a',v1);

dbms\_output.put\_line(v1);

end;

10) Create a table Emp\_Details which has the following specifications.

Name Data type Constraint

Empno Integer Primary Key

Membership\_no Char Unique

Name Varchar Not Null

Salary Integer Above 10000

Mgr Integer Refers to the Empno column

Take all the column values from the user and enter the record in the table Emp\_Details. If any constraint violation happens then provide user friendly error message according to the error.

ANS: declare

DUPLICATE\_empno1 exception;

Checkk exception;

NOT\_NULL exception;

Foreign\_key exception;

pragma exception\_init(DUPLICATE\_empno1,-00001);

pragma exception\_init(Checkk,-02290);

pragma exception\_init(NOT\_NULL,-01400);

pragma exception\_init(Foreign\_key,-02291);

begin

insert into employeeno

values(&a,'&b','&c',&d,&e);

exception WHEN DUPLICATE\_empno1 then

dbms\_output.put\_line('Not unique');

WHEN Checkk then

dbms\_output.put\_line('Not greater than 10000');

WHEN NOT\_NULL then

dbms\_output.put\_line('Not null');

when Foreign\_key then

dbms\_output.put\_line('parent key not found');

end;

11) Create a table Emp\_Proc having structure and records same as that of emp table.

Create a procedure Delete\_Employee that will take department name as a parameter. The procedure should delete the record of the employees from the Emp\_EmployeeID table only for the department name passed.

ANS: drop table emp\_rec

create table emp\_rec

as

select \* from emp

create or replace procedure Delete\_Employee(p\_dname varchar2)

is

begin

delete from emp\_rec

where deptno= (select deptno from dept where dname=p\_dname);

dbms\_output.put\_line('RECORD DELETED');

end;

begin

Delete\_Employee('ACCOUNTING');

end;

12) Create a table Company\_Data with columns empid, ename and job. There is no primary key or unique key constraint to the empid column.

Create a procedure Add\_Company\_Data which will take empid as the parameter and adds a new record in that table if the empid with the same value does not exist.

ANS:

drop table company\_data

create table company\_data

(empid number(4),

ename varchar2(30),

job varchar2(20));

create or replace procedure add\_data(empn in number) is

flag number(2):=0;

create cursor c1 Is select empid from company\_data;

begin

for em in c1 then

if em=empn then

flag:=1;

end if;

end loop;

if flag=0 then

insert into company\_data values(empn,'asdf','sadf');

dbms\_output.put\_line('record successfully inserted');

else

dbms\_output.put\_line('Record Duplicacy. Record not inserted');

end if;

end;

show error;

begin

add\_data(10);

end;

13) Considering the emp table create a procedure IsHighest that will take ename as the parameter.The procedure should reply whether employee is the highest earner or not.

14) Create a table student. It has four fields rollno, name, marks and grade.

Note – Grade column has be generated by a function get\_Grade(). This function takes marks as argument and generates the grade as per the following table

Marks Criteria Grade

Greater than or equal to 70 Distinction

Between 60 and 69 First Class

Between 50 and 59 Second Class

Less than 50 Failed.

When the record is inserted then the grade value should be generated.

15) Create a function Yearly\_Raise that will take the salary, deptno and job as the parameters and raise the salary according to different criteria.

Criteria Raise

Clerk employees of deptno 20 earning salary above 1000 20%

Clerk employees of deptno 20 earning salary less 1000 15%

Clerk employees of deptno 20 earning salary above 1000 25%

Clerk employees of deptno 20 earning salary less than 1000 18%

Clerk employees of deptno 30 having any salary 10%

Use this function to update salaries of the employees of job Clerk in the table emp.

16)Create a function named USER\_VALID\_DEPTNO that has a single parameter p\_dno to accept a department number and returns a BOOLEAN value. The function returns TRUE if the department number exists in the DEPARTMENTS table else it returns FALSE.

• Create a procedure named SHOW\_STRENGTH that accepts department number in a single parameter p\_deptno from user. The procedure gives a call to USER\_VALID\_DEPTNO. If the function returns TRUE then the procedure finds out how many employees are there in the department from the EMPLOYEES table and displays the same on the screen. If the function returns FALSE then the procedure displays an appropriate error message.

• Give call to SHOW\_STRENGTH by passing on department number 10. Do the same for department number 76

17) Create a package Emp\_Package that will have a plsql table called Emp\_Info containing Enames and Sal from the emp table.

The package has two procedure Show\_Clerks and Show\_Managers which will display the enames and salaries of the Clerks and Managers respectively. These procedures should take the names and salaries from the Emp\_Info plsql table only.

The Emp\_Info always should get loaded within the session first when any of the package member is getting called for the first time within that session.

18) Create a trigger DML\_on\_View which will allow insert, update and delete for a view which is based on an inner join between tables Emp and Dept