PL/SQL Lab Assignment-3

1. Write a PL/SQL program to demonstrate following exceptions:

• *Too Many Rows*

• *No Data Found*

STATEMENT:

```
CREATE TABLE EMP(ENO number(5) Primary key,ENAME VARCHAR(15),
JOB VARCHAR(20), DEPTNO NUMBER(5));

Insert into emp values(1,'Surjit','Prof',10);

Insert into emp values(2,'Ranjit','Ass.Prof',20);

Insert into emp values(3,'Manjit','Prof',10);

declare

name emp.ename%type;

begin

select ename into name from emp where deptno=30;

dbms_output.put_line(name);

exception

when TOO_MANY_ROWS then

dbms_output.put_line('too amny rows exception is caught');

when NO_DATA_FOUND then

dbms_output.put_line('no data found exception is caught');

end;
```

OUTPUT:

```
Statement processed.
no data found exception is caught
```

2. Write a PL/SQL code to display a message to check whether the record is deleted or not.

STATEMENT:

```
begin
delete from emp where deptno=30;
if sql%found then
dbms_output.put_line('Delete operation is performed successfully!');
else
dbms_output.put_line('Delete operation not performed successfully!');
end if;
end;
```

OUTPUT:

```
Statement processed.
Delete operation not performed successfully!
```

3. Write a PL/SQL code to display a message to provide the information about the number of records deleted by the delete statement issued in a PL/SQL block.

EMP (empno, ename, job, sal, deptno)

STATEMENT:

```
begin
delete from emp where deptno=20;
dbms_output.put_line(sql%rowcount ||' row have been deleted...');
end;
```

OUTPUT:

```
Statement processed.
1 row have been deleted...
```

4. Write a PL/SQL code to display the empno, ename, job of employees of department number 10 for EMP table.

STATEMENT:

```
declare
cursor c1 is select * from emp where deptno=10;
rec emp%ROWTYPE;
begin
open c1;
loop
fetch c1 into rec;
exit when c1%notfound;
dbms_output.put_line('Employee no. : '|| rec.eno);
dbms_output.put_line('Employee name : '|| rec.ename);
dbms_output.put_line('Job : '|| rec.job);
```

```
dbms_output.put_line('  ');

end loop;

close c1;

end;
```

OUTPUT:

```
Statement processed.
Employee no. : 1
Employee name : Surjit
Job : Prof

Employee no. : 3
Employee name : Manjit
Job : Prof
```

5. Write a PL/SQL code to display the employee number and name of top 5 highest paid employees.

Let us consider CUST table with columns Cno PK, MeterNo Unique, Prev_Reading, Current_Reading, Units, Bill_Amount Units=Current_Reading-Prev_Reading

• Bill will be charged according to following rate:

• First 100 Units 0.5 Rs per unit and after 0.75 Rs for every unit consume

Cno	MeterNo	Prev_Reading	Current_Reading	Units	Bill_Amount
89	100	300	800		
67	101	790	1000		
90	200	800	1200		
62	789	200	800		
70	899	3200	8700		
66	500	3000	9800		

STATEMENT:

```
CREATE TABLE cust (
        cno number(10) Primary key,
        meterno number(10),
        prev_reading number(10),
        current_reading number(10),
        units number(10) ,
        bill_amount number(20)
);
insert into cust values(89,100,300,800,null,null);
insert into cust values(67,101,790,1000,null,null);
insert into cust values(90,200,800,1200,null,null);
insert into cust values(62,789,200,800,null,null);
insert into cust values(70,899,3200,8700,null,null);
insert into cust values(66,500,3000,9800,null,null);
declare
rec cust%ROWTYPE;
pr number;
cr number;
u number;
ba number;
```

6. Write a PL/SQL code to update units and bill amount of customer number 89 in CUST table

STATEMENT:

```
begin
select current_reading,prev_reading into cr,pr from cust where cno=89;
u:=cr-pr;
update cust set units=u where cno=89;
if u>100 then
ba:= (100*0.5) + ((u-100)*0.75);
else
ba:= u*0.5;
end if;
update cust set bill_amount=ba where cno=89;
end;
SELECT * from cust;
```

OUTPUT:

CNO	METERNO	PREV_READING	CURRENT_READING	UNITS	BILL_AMOUNT
89	100	300	800	500	350
67	101	790	1000	-	-
90	200	800	1200	-	-
62	789	200	800	-	-
70	899	3200	8700	-	-
66	500	3000	9800	-	-

7. Write a PL/SQL code to update units and bill amount of all the customers in CUST table

STATEMENT:

```
declare
rec cust%ROWTYPE;
cursor c1 is select cno,current_reading,prev_reading from cust;
u number;
ba number;
begin
for rec in c1 loop
u:=rec.current_reading-rec.prev_reading;
update cust set units=u where cno=rec.cno;
if u>100 then
ba:= (100*0.5) + ((u-100)*0.75);
else
ba:= u*0.5;
end if;
update cust set bill_amount=ba where cno=rec.cno;
end loop;
end
SELECT * from cust;
```

OUTPUT:

CNO	METERNO	PREV_READING	CURRENT_READING	UNITS	BILL_AMOUNT
89	100	300	800	500	350
67	101	790	1000	210	133
90	200	800	1200	400	275
62	789	200	800	600	425
70	899	3200	8700	5500	4100
66	500	3000	9800	6800	5075

Download CSV