National Institute of Technology, Tiruchirapalli

Department of Computer Applications

DBMS LAB MANUAL

SUBMITTED BY:-

NAME – *KULDEEP PATIDAR*

ROLL NO. – *205119047*

SECTION - A

INDEX

EXERCISE.1

1.1

```
CREATE TABLE EMP (EMPNO INTEGER PRIMARY KEY,ENAME VARCHAR(20) NOT NULL,JOB
VARCHAR(20) NOT NULL,MGR INTEGER,DEPTNO INTEGER,SAL INTEGER);
```

1.2

```
ALTER TABLE EMP ADD COMM INTEGER ;
```

1.3

```
ALTER TABLE EMP MODIFY JOB VARCHAR(30);
```

1.4

```
CREATE TABLE DEPT(DEPTNO INTEGER PRIMARY KEY,DNAME VARCHAR(20),LOC VARCHAR(40));
```

1.5

```
ALTER TABLE EMP ADD FOREIGN KEY (DEPTNO) REFERENCES DEPT(DEPTNO);
```

1.6

```
ALTER TABLE EMP ADD CHECK (EMPNO>100);
```

1.7

```
ALTER TABLE EMP modify sal integer default 5000;
```

1.8

```
ALTER TABLE EMP ADD DOB VARCHAR(10);
```

EXERCISE.2

2.1

```
INSERT INTO DEPT VALUES(10, 'MANAGEMENT','MAIN BLOCK');

INSERT INTO DEPT VALUES(20, 'DEVELOPMENT','MANUFACTURING');

INSERT INTO DEPT VALUES(30, 'MAINTAINANCE','UNIT MAN BLOCK');

INSERT INTO DEPT VALUES(40, 'TRANSPORT','ADMIN BLOCK');

INSERT INTO DEPT VALUES(50, 'SALES','HEAD OFFICE');
```

2.2

```
INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7369,'SMITH','CLERK',7566,'17-DEC80',800,0,20);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7399,'ASANT','SALESMAN',7566,'20-FEB81',1600,300,20);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7499,'ALLEN','SALESMAN',7698,'20-FEB81',1600,300,30);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7521,'WARD','SALESMAN',7698,'22-FEB82',1250,500,30);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7566,'JONES','MANAGER',7839,'02-APR81',5975,500,20);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7698,'BLAKE','MANAGER',7839,'01-MAY79',9850,1400,30);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL , DEPTNO)
VALUES(7611,'SCOTT','HOD',7839,'12-JUN76',3000,10);

INSERT INTO EMP(EMPNO, ENAME ,JOB ,DOB ,SAL , DEPTNO) VALUES(7839,'CLARK','CEO','16-
MAR72',9900,10);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7368,'FORD','SUPERVIS',7366,'17-DEC80',800,0,20);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7599,'ALLEY','SALESMAN',7698,'20-FEB81',1600,300,30);

INSERT INTO EMP(EMPNO, ENAME ,JOB, MGR ,DOB ,SAL ,COMM, DEPTNO)
VALUES(7421,'DRANK','CLERCK',7698,'22-JAN82',1250,500,30);
```

2.3

```
UPDATE EMP SET COMM=1000 WHERE JOB='MANAGER';
```

2.4

```
CREATE TABLE EMPLOYEE (EMPNO INTEGER PRIMARY KEY,ENAME VARCHAR(20) NOT NULL,JOB
VARCHAR(30) NOT NULL,MGR INTEGER,DEPTNO INTEGER,SAL INTEGER,COMM INTEGER,DOB
VARCHAR(10));

INSERT INTO EMPLOYEE SELECT*FROM EMP;
```

2.5

```
DELETE FROM EMPLOYEE WHERE JOB='SUPERVIS';
```

2.6

```
DELETE FROM EMPLOYEE WHERE EMPNO=7599;
```

2.7

```
SELECT * FROM EMP ORDER BY SAL;
```

2.8

```
SELECT * FROM EMP ORDER BY SAL DESC;
```

2.9

```
SELECT * FROM EMP WHERE DEPTNO=30;
```

2.10

```
SELECT DISTINCT DEPTNO FROM EMP;
```

2.11

```
SELECT * FROM EMP ORDER BY ENAME;
```

2.12

```
create table manager as select * from EMP where JOB='MANAGER';
```

2.13

```
select * from EMP where COMM=NULL ;
```

2.14

```
select ENAME,DNAME from EMP,DEPT where EMP.DEPTNO=DEPT.DEPTNO ;
```

EXERCISE.3

3.1

```
select * from EMP where DEPTNO in(7369,7499);
```

3.2

```
select * from EMPLOYEE where ENAME like "S%";
```

3.3

```
select * from EMPLOYEE where ENAME not like "S%";
```

3.4

```
select * from EMPLOYEE where EMPNO between 7500 and 7600 ;
```

3.5

```
Select * from EMPLOYEE where EMPNO not between 7500 and 7600 ;
```

3.6

```
select sqrt(SAL) from EMP;
```

3.7

```
SELECT COUNT(*) FROM EMP;
```

3.8

```
SELECT SUM(SAL),AVG(SAL) FROM EMP;
```

3.9

```
select min(SAL) "MIN_SAL", MAX(SAL) "MAX_SAL" from EMP;
```

3.10

```
SELECT SUM(SAL) FROM EMP;
```

3.11

```
SELECT JOB,SUM(SAL) FROM EMP GROUP BY JOB;
```

3.12

```
select to_date(DOB,'DD-MM-YY') from EMP;
```

3.13

```
select add_months(DOB,2) from EMP;
```

3.14

```
select last_day('05-oct-09') from dual;
```

3.15

```
select round(to_date(dob),'month') from emp;
```

3.16

```
select round(to_date(dob),'year') from emp;
```

3.17

```
select round(to_date(dob),'day') from emp;
```

3.18

```
select(sysdate-60) from dual;*/
```

3.19

```
select ENAME ,SAL , SAL+0.15* SAL from EMP;
```

3.20

```
select ENAME from EMP where ENAME like 'B%' or ENAME like 'C%';
```

3.21

```
select ENAME,SAL,MGR from EMP where SAL in (select min(SAL) from EMP group by MGR);
```

3.22

```
select dname, count (ename) from emp, dept where emp.deptno=dept.deptno group by
dname
```

3.23

```
select ename from emp where length (empname) <=5;
```

3.24

```
select ename from emp where mgr in(7602,7566,7789);
```

3.25

```
select count (distinct job) from emp;
```

3.26

```
select max(sal)-min(sal) from emp;
```

3.27

```
select count(distinct deptno) from emp;
```

3.28

```
select empname , dob from emp where to_char (dob,'MON')='FEB';
```

3.29

```
select ENAME from EMP where ENAME LIKE ('S%') and ENAME LIKE('%H');
```

3.30

```
select ename from emp where sal>5000 or sal>6000;
```

<u>EXERCISE.4</u>

```
select ENAME,DNAME from EMP,DEPT where DNAME='MAINTAINANCE' OR
DNAME='DEVELOPMENT' ;
```

```
SELECT  ename FROM emp WHERE sal >(SELECT MIN(saL)FROM emp) AND JOB LIKE ('M%');
```

```
SELECT ename FROM EMP WHERE job =( SELECT job FROM emp WHERE eNAME='JONES');
SELECT  *  FROM emp WHERE sal >ANY( SELECT sal FROM emp WHERE DEPTNO=30 );
```

```
SELECT * FROM EMP WHERE job =( SELECT job FROM emp WHERE eNAME='JONES') AND SAL>=(
SELECT sal FROM emp WHERE ENAME='FORD');
```

```
SELECT ename, job FROM emp WHERE DEPTNO=10 AND  JOB IN(SELECT JOB FROM emp,dept
WHERE EMP.DEPTNO=DEPT.DEPTNO AND Dname='MANAGEMENT');
```

```
SELECT * FROM emp WHERE sal >(SELECT AVG(SAL)FROM emp);
```

```
SELECT ENAME,JOB,DNAME FROM EMP,DEPT WHERE EMP.DEPTNO=DEPT.DEPTNO;
```

```
SELECT * FROM EMP WHERE job in (SELECT job FROM emp,dept WHERE emp.deptno=dept.deptno
and LOC='MAIN BLOCK');
```

```
SELECT * FROM emp WHERE DEPTNO=10 AND  JOB IN(SELECT JOB FROM emp,dept WHERE
EMP.DEPTNO=DEPT.DEPTNO AND Dname='development');
```

```
SELECT * FROM EMP WHERE job =( SELECT job FROM emp WHERE eNAME='FORD') AND SAL=(
SELECT SAL FROM emp WHERE eNAME='FORD');
```

```
SELECT  *  FROM emp WHERE deptno=20 and job=ANY( SELECT job FROM emp WHERE DEPTNO=30
);
```

```
SELECT eNAME FROM emp WHERE sal >ANY( SELECT sal FROM emp WHERE DEPTNO IN (20,30));
```

select ename,dname from emp left join dept on emp.deptno=dept.deptno;

select ename,dname from emp right join dept on emp.deptno=dept.deptno;

select ename,dname from emp full outer join dept on emp.deptno=dept.deptno;

select ename,job,dname,loc from emp natural join dept;

EXERCISE.5

select deptno from dept union select deptno from accdept;

select deptno from dept union all select deptno from accdept;

select deptno from dept intersect select deptno from accdept;

select deptno from dept minus select deptno from accdept;

create view managers as select * from employee where job='manager';

create view emps as select empno,ename,employee.deptno,dept.dname from employee,dept where employee.deptno=dept.deptno;

create view emps2 as select empno,ename,employee.deptno,dept.dname from employee,dept where employee.deptno=dept.deptno and job not in ('hod','ceo');

SHOW FULL TABLES
WHERE table_type = 'VIEW';

drop view managers;

EXERCISE.6

Program 6.1:write a pl/sql program to swap two numbers with out taking third variable

```
declare
a number(10);
b number(10);
begin
a:=&a;
b:=&b;
dbms_output.put_line('THE PREV VALUES OF A AND B WERE');
dbms_output.put_line(a);
dbms_output.put_line(b);
a:=a+b;
b:=a-b;
a:=a-b;
dbms_output.put_line('THE VALUES OF A AND B ARE');
dbms_output.put_line(a);
dbms_output.put_line(b);
end;
```

OUTPUT:

```
SQL> @ SWAPPING.SQL
17 /
Enter value for a: 5
old 5: a:=&a;
new 5: a:=5;
Enter value for b: 3
old 6: b:=&b;
```

```
new 6: b:=3;
THE PREV VALUES OF A AND B WERE
5
3
THE VALUES OF A AND B ARE
3
5
PL/SQL procedure successfully completed.
```

Program 6.2:write a pl/sql program to swap two numbers by taking third variable

```
declare
a number(10);
b number(10);
c number(10);
begin
dbms_output.put_line('THE PREV VALUES OF A AND B WERE');
dbms_output.put_line(a);
dbms_output.put_line(b);
a:=&a;
b:=&b;
c:=a;
a:=b;
b:=c;
dbms_output.put_line('THE VALUES OF A AND B ARE');
dbms_output.put_line(a);
dbms_output.put_line(b);
end;
```

OUTPUT:

```
SQL> @ SWAPPING2.SQL
19 /
Enter value for a: 5
```

```
old 6: a:=&a;

new 6: a:=5;

Enter value for b: 3

old 7: b:=&b;

new 7: b:=3;

THE PREV VALUES OF A AND B WERE

5

3

THE VALUES OF A AND B ARE

3

5

PL/SQL procedure successfully completed.
```

Program 6.3:

Write a pl/sql program to find the largest of two numbers

```
declare

a number;

b number;

begin

a:=&a;

b:=&b;

if a=b then

dbms_output.put_line('BOTH ARE EQUAL');

elsif a>b then

dbms_output.put_line('A IS GREATER');

else

dbms_output.put_line('B IS GREATER');

end if;

end;
```

OUTPUT:

```
SQL> @ GREATESTOF2.sql
```

```
13 /
Enter value for a: 5
old 5: a:=&a;
new 5: a:=5;
Enter value for b: 2
old 6: b:=&b;
new 6: b:=2;
A IS GREATER
PL/SQL procedure successfully completed.
```

Program 6.4:write a pl/sql program to find the total and average of 6 subjects and display the grade

```
declare
java number(10);
dbms number(10);
co number(10);
se number(10); es
number(10); ppl
number(10); total
number(10); avgs
number(10); per
number(10);
begin
dbms_output.put_line('ENTER THE MARKS');
java:=&java;
dbms:=&dbms;
co:=&co;
se:=&se;
es:=&es;
ppl:=&ppl;
total:=(java+dbms+co+se+es+ppl);
```

```
per:=(total/600)*100;

if java<40 or dbms<40 or co<40 or se<40 or es<40 or ppl<40 then

dbms_output.put_line('FAIL');

if per>75 then

dbms_output.put_line('GRADE A');

elsif per>65 and per<75 then

dbms_output.put_line('GRADE B');

elsif per>55 and per<65 then

dbms_output.put_line('GRADE C');

else

dbms_output.put_line('INVALID INPUT');

end if;

dbms_output.put_line('PERCENTAGE IS '||per);

dbms_output.put_line('TOTAL IS '||total);

end;
```

OUTPUT:

```
SQL> @ GRADE.sql

31 /

Enter value for java: 80

old 12: java:=&java;

new 12: java:=80;

Enter value for dbms: 70

old 13: dbms:=&dbms;

new 13: dbms:=70;

Enter value for co: 89

old 14: co:=&co;

new 14: co:=89;

Enter value for se: 72

old 15: se:=&se;

new 15: se:=72;

Enter value for es: 76
```

```
old 16: es:=&es;

new 16: es:=76;

Enter value for ppl: 71

old 17: ppl:=&ppl;

new 17: ppl:=71;

GRADE A

PERCENTAGE IS 76

TOTAL IS 458

PL/SQL procedure successfully completed.
```

Program 6.5:

Write a pl/sql program to find the sum of digits in a given number

```
declare
a number;
d number:=0;
sum1 number:=0;
begin
a:=&a;
while a>0
loop
d:=mod(a,10);
sum1:=sum1+d;
a:=trunc(a/10);
end loop;
dbms_output.put_line('sum is'|| sum1);
end;
```

OUTPUT:

```
SQL> @ SUMOFDIGITS.sql
16 /
```

Program 6.6:write a pl/sql program to display the number in reverse order

```
declare
a number;
rev number;
d number;
begin
a:=&a;
rev:=0;
while a>0
loop
d:=mod(a,10);
rev:=(rev*10)+d;
a:=trunc(a/10);
end loop;
dbms_output.put_line('no is'|| rev);
end;
```

OUTPUT:

```
SQL> @ REVERSE2.sql
16 /
Enter value for a: 536
old 6: a:=&a;
new 6: a:=536;
no is635
PL/SQL procedure successfully completed.
```

Program 6.7:

Write a pl/sql program to check whether the given number is prime or not

```
declare
a number;
c number:=0;
i number;
begin
```

```
a:=&a;
for i in 1..a
loop
if mod(a,i)=0 then
c:=c+1;
end if;
end loop;
if c=2 then
dbms_output.put_line(a ||'is a prime number');
else
dbms_output.put_line(a ||'is not a prime number');
end if;
end;
```

OUTPUT:

```
SQL> @ PRIME.SQL
19 /
Enter value for a: 11
old 6: a:=&a;
new 6: a:=11;
11is a prime number
PL/SQL procedure successfully completed.
```

Program 6.8:

Write a pl/sql program to find the factorial of a given number

```
declare
n number;
f number:=1;
begin
n:=&n;
for i in 1..n
loop
```

```
f:=f*i;
end loop;
dbms_output.put_line('the factorial is'|| f);
end;
```

OUTPUT:

```
SQL> @ FACTORIAL.sql
12 /
Enter value for n: 5
old 5: n:=&n;
```

Program 6.9:write a pl/sql code block to calculate the area of a circle for a value of radius varying from 3 to 7.

Store the radius and the corresponding values of calculated area in an empty table named areas ,consisting of two columns radius & area

TABLE NAME:AREAS

RADIUS AREA

```
SQL> create table areas(radius number(10),area number(6,2));
Table created.
--PROGRAM
declare
pi constant number(4,2):=3.14;
radius number(5):=3;
area number(6,2);
begin
while radius<7 loop
area:=pi*power(radius,2);
insert into areas values(radius,area);
radius:=radius+1;
end loop;
end;
```

OUTPUT:

```
SQL> @ AREAOFCIRCLE.SQL

13 /

PL/SQL procedure successfully completed.

SQL> SELECT * FROM AREAS;

RADIUS AREA

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

3 28.26

4 50.24

5 78.5

6 113.04
```

Program 6.10:write a pl/sql code block that will accept an account number from the user,check if the users balance is less than minimum balance,only then deduct rs.100/- from the balance.this process is fired on the acct table.

```
SQL> create table acct(name varchar2(10),cur_bal number(10),acctno number(6,2));

SQL> insert into stud values('&sname',&rollno,&marks);

SQL> select * from acct;

ACCTNO NAME CUR_BAL

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

777 sirius 10000

765 john 1000

855 sam 500

353 peter 800

--PROGRAM

declare

mano number(5);

mcb number(6,2);

minibal constant number(7,2):=1000.00;

fine number(6,2):=100.00;

begin

mano:=&mano;
```

```
select cur_bal into mcb from acct where acctno=mano;
if mcb<minibal then
update acct set cur_bal=cur_bal-fine where acctno=mano;
end if;
end;
```

OUTPUT:

```
SQL> @ BANKACC.sql
13 /
Enter value for mano: 855
old 7: mano:=&mano;
new 7: mano:=855;
PL/SQL procedure successfully completed.
```

<u>EXERCISE.7</u>

7.1 create or replace procedure salary(deptid number) as

```
begin
    update emp set sal=sal+1000 where sal>5000 AND deptno=deptid;
end;
```

7.2 create or replace procedure salary1(empid number) as

```
begin
    update emp set sal=sal+sal*(0.1) where empno=empid;
end;
```

7.3 create or replace procedure get_sal(dept number) as

```
begin
     for s in (select * from emp where deptno = dept)
     loop
       dbms_output.put_line(s.sal);
     end loop;
  end;
```

7.4 create or replace procedure get_nature(dept number) as

```
begin
```

```
          for s in (select * from emp where deptno = dept)
          loop
             dbms_output.put_line(s.job);
           end loop;
        end;
7.5   create or replace procedure dep_name(deptid number)  as
       begin
         select dept.dname from dept,emp where emp.deptno=dept.deptno;
       end;
```

EXERCISE.8

8.1

```
CREATE OR RELPLACE TRIGGER trig1 before insert on DEPT for each row DECLARE a number;
BEGIN
        if(:new.DEPTNO is Null) then
                raise_application_error(-20001,'error:: DEPTNO cannot be null');
        else
                select count(*) into a from DEPT where DEPTNO =:new.DEPTNO;
                if(a=1) then
                        raise_application_error(-20002,'error:: cannot have duplicate
                DEPTNo ');
                end if;
        end if;
END;
```

8.2

```
CREATE [OR REPLACE] TRIGGER trig2 After delete on DEPT FOR EACH ROW
BEGIN
        DELETE FROM emp WHERE emp.deptno=:new.deptno;
```

```
END;
```

8.3

```
CREATE TRIGGER trig3 AFTER DELETE ON emp FOR EACH ROW
BEGIN
        INSERT INTO log(val1, val2, ...) VALUES (old.val1, old.val2, ...);

  END;
```