Department of Computer Applications

Roll No. – 205119051

Name – Lakshay

Subject – CA 702

Database Systems Lab

1. <u>Data Definition Language (DDL) Commands</u>

<u>1.1</u> Create table EMPLOYEE(EMPNO number(4), ENAME varchar2(25),

JOB varchar2(15), MGR number(4), DEPTNO number(3),

SALARY number(7,2));

<u>1.2</u> Alter table EMPLOYEE add COMM number(5);

<u>1.3</u> Alter table EMPLOYEE modify JOB varchar2(20);

<u>1.4</u> Create table DEPARTMENT(DEPTNO number(2), DNAME varchar2(10), DLOC varchar2(20));

<u>1.5</u> Alter table EMPLOYEE modify EMPNO NOT NULL PRIMARY KEY;

Alter table DEPARTMENT add FOREIGN KEY(DEPTNO) REFERENCES DEPARTMENT(DEPTNO);

<u>1.6</u> Alter table EMPLOYEE add CHECK(EMPNO>100);

<u>1.7</u> Alter table EMPLOYEE modify SALARY DEFAULT 5000 NOT NULL;

<u>1.8</u> Alter table EMPLOYEE add DOB date;

2. Data Manipulation Language (DDL) Commands

2.1 Insert all

```
  into DEPARTMENT(DNAME, DEPTNO, DLOC) values
            (10, ’MANAGEMENT’, ’MAIN BLOCK’)
        into DEPARTMENT (DNAME, DEPTNO, DLOC) values
            (20, ’DEVELOPMENT’,’MANUFACTURING UNIT’)
        into DEPARTMENT (DNAME, DEPTNO, DLOC) values
            (30, ’MAINTAINANCE’, ’MAIN BLOCK’)
        into DEPARTMENT (DNAME, DEPTNO, DLOC) values
            (40, ’TRANSPORT’, ’ADMIN BLOCK’)
        into DEPARTMENT (DNAME, DEPTNO, DLOC) values
            (50, ’SALES’, ’HEAD OFFICE’)
      Select * from DUAL;
```

2.2 insert into EMPLOYEE values(7369,'Smith','Clerk',7566,20,800,0,'17-dec-1980');

```
      insert into EMPLOYEE values(7399,'Asant','Salesman',7566,20,1600,300,'20-feb-1981');

     insert into EMPLOYEE values(7499,'Allen','Salesman',7698,30,1600,300,'20-feb-1981');

     insert into EMPLOYEE values(7521,'Warl','Salesman',7698,30,1250,500,'22-feb-1982');

     insert into EMPLOYEE values(7566,'Jones','Manager',7839,20,5975,500,'02-apr-1981');

     insert into EMPLOYEE values(7698,'Blake','Manager',7839,30,9850,1400,'01-may-1979');
```

insert into EMPLOYEE values(7611,'Scott','HOD',7839,10,3000,NULL,'12-jun-1976');

insert into EMPLOYEE values(7839,'Clark','CEO',NULL,10,9900,NULL,'16-mar-1972');

insert into EMPLOYEE values(7368,'Ford','Supervisor',7366,20,800,0,'17-dec-1980');

insert into EMPLOYEE values(7599,'Alley','Salesman',7698,30,1600,300,'20-feb-1981');

insert into EMPLOYEE values(7421,'Drank','Clerck',7698,30,1250,500,'22-jan-1982');

<u>2.3</u> Update EMPLOYEE set COMM=1000 where JOB='manager';

<u>2.4</u> Create table EMPLOYEE2 (EMPNO number(6), ENAME varchar2(20),

JOB varchar2(10), MGR number(4), DEPTNO number(3),

SALARY number(7,2),COMM number(5), DOB date);

INSERT INTO EMPLOYEE2 SELECT * FROM EMPLOYEE;

<u>2.5</u> Delete from EMPLOYEE2 where JOB='Supervisor';

<u>2.6</u> Delete from EMPLOYEE2 where EMPNO=7599;

<u>2.7</u> Select * from EMPLOYEE ORDER BY SALARY;

<u>2.8</u> Select * from EMPLOYEE ORDER BY SALARY DESC;

<u>2.9</u> Select * from EMPLOYEE where DEPTNO=30;

<u>2.10</u> Select DISTINCT DEPTNO from EMPLOYEE;

<u>2.11</u> Select * from EMPLOYEE ORDER BY ENAME;

<u>2.12</u> create table MANAGER as Select * from EMPLOYEE where JOB='Manager';

<u>2.13</u> Select * from EMPLOYEE where COMM=NULL;

<u>2.14</u> Select ENAME,DNAME from EMPLOYEE, DEPARTMENT where EMPLOYEE.DEPTNO=DEPARTMENT.DEPTNO;

3. <u>In-Built Functions</u>

<u>3.1</u> Select * from EMPLOYEE where DEPTNO IN (20,10);

<u>3.2</u> Select * from EMPLOYEE where ENAME LIKE 'S%';

<u>3.3</u> Select * from EMPLOYEE where ENAME NOT LIKE ‘S%’;

<u>3.4</u> Select * from EMPLOYEE where EMPNO BETWEEN 7500 AND 7600;

<u>3.5</u> Select * from EMPLOYEE where EMPNO NOT BETWEEN 7500 AND 7600;

<u>3.6</u> Select SQRT(SALARY) from EMPLOYEE;

<u>3.7</u> Select COUNT(*) from EMPLOYEE;

<u>3.8</u> Select SUM(SALARY), AVG(SALARY) from EMPLOYEE;

<u>3.9</u> Select MIN(SALARY) as MIN_SALARY, MAX(SALARY) as MAX_SALARY from EMPLOYEE;

<u>3.10</u> Select SUM(SALARY) from EMPLOYEE;

<u>3.11</u> Select JOB, SUM(SALARY) from EMPLOYEE GROUP BY JOB;

<u>3.12</u> Select TO_CHAR(TO_DATE(’14-jul-09’),’month’) from DUAL;

<u>3.13</u> Select TO_DATE(DOB,’DD-MM-YY’) from EMPLOYEE;

<u>3.14</u> Select ADD_MONTHS(DOB,2) from EMPLOYEE;

<u>3.15</u> Select LAST_DAY(’05-oct-09’) from DUAL;

<u>3.16</u> Select ROUND(TO_DATE(DOB),'month') from EMPLOYEE;

Select ROUND(TO_DATE(DOB),'year') from EMPLOYEE;

Select ROUND(TO_DATE(DOB),'day') from EMPLOYEE;

<u>3.17</u> Select (SYSDATE-60) from DUAL;

<u>3.18</u> Select ENAME , SALARY, SALARY+0.15*SALARY from EMPLOYEE;

<u>3.19</u> Select ENAME from EMPLOYEE where ENAME LIKE ‘B%’ or ENAME LIKE ‘C%’;

<u>3.20</u> Select ENAME, SALARY, MGR, from EMPLOYEE where SALARY in (Select MIN(SALARY) from EMPLOYEE GROUP BY MGR);

<u>3.21</u> Select DNAME, COUNT(ENAME) from EMPLOYEE, DEPARTMENT where EMPLOYEE.DEPTNO=DEPARTMENT.DEPTNO GROUP BY DNAME;

<u>3.22</u> Select ENAME from EMPLOYEE where length(ENAME)<=5;

<u>3.23</u> Select ENAME from EMPLOYEE where MGR IN(7399,7698,7566);

<u>3.24</u> Select COUNT(DISTINCT JOB) from EMPLOYEE;

<u>3.25</u> Select MAX(SALARY) - MIN(SALARY) from EMPLOYEE;

<u>3.26</u> Select COUNT(DISTINCT DEPTNO) from EMPLOYEE;

<u>3.27</u> Select ENAME,DOB from EMPLOYEE where TO_CHAR(DOB,’MON’)=’feb’;

<u>3.28</u> Select ENAME from EMPLOYEE where TO_CHAR(DOB,’MON’) LIKE TO_CHAR(SYSDATE,’MON’);

<u>3.29</u> Select ENAME from EMPLOYEE where ENAME LIKE ‘S%H’;

<u>3.30</u> Select ENAME from EMPLOYEE where SALARY>5000;

4. <u>Nested Queries and Joins</u>

<u>4.1</u> Select * from EMPLOYEE, DEPARTMENT where EMPLOYEE.DEPTNO=DEPARTMENT.DEPTNO and (DEPARTMENT.DNAME='MAINTAINANCE' orDEPT.DNAME='DEVELOPMENT')

<u>4.2</u> Select ENAME,SALARY from EMPLOYEE where SALARY>(Select MIN(SALARY) from EMPLOYEE) and JOB LIKE 'M%';

<u>4.3</u> Select * from EMPLOYEE where JOB=(Select JOB from EMPLOYEE where ENAME='jones');

<u>4.4</u> Select * from EMPLOYEE where SALARY>(Select MAX(SALARY) from EMPLOYEE where DEPTNO=30);

<u>4.5</u> Select * from EMPLOYEE where JOB=(Select JOB from EMPLOYEE where ENAME='jones') and SALARY>=(Select SALARY from EMPLOYEE where ENAME='ford');

<u>4.6</u> Select ENAME, JOB from EMPLOYEE where DEPTNO=20 and JOB=ANY(Select JOB from EMPLOYEE E, DEPARTMENT D where E.DEPTNO=D.DEPTNO and D.DNAME='MANAGEMENT');

<u>4.7</u> Select * from EMPLOYEE OUT where SALARY>(Select AVG(SALARY) from EMPLOYEE where DEPTNO=OUT.DEPTNO);

<u>4.8</u> Select ENAME, JOB, DNAME from EMPLOYEE E, DEPARTMENT D where E.DEPTNO=D.DEPTNO;

<u>4.9</u> Select * from EMPLOYEE where JOB=ANY(Select E.JOB from DEPARTMENT D,EMPLOYEE E where D.DEPTNO=E.DEPTNO and DLOC='MAIN BLOCK') and DEPTNO!=(Select DEPTNO from DEPARTMENT where DLOC='MAIN BLOCK');

<u>4.10</u> Select * from EMPLOYEE where DEPTNO=10 and JOB=ANY(Select JOB from EMPLOYEE, DEPARTMENT where DEPARTMENT.DEPTNO=EMPLOYEE.DEPTNO and DEPARTMENT.DNAME='DEVELOPMENT');

<u>4.11</u> Select * from EMPLOYEE where JOB=(Select JOB from MEP where ENAME='ford') and SALARY=(Select SALARY from EMPLOYEE where ENAME='ford');

4.12 Select DNAME from DEPARTMENT where DEPTNO= ANY(Select DEPTNO from (Select COUNT(JOB) as NO, DEPTNO from EMPLOYEE where JOB='salesman' GROUP BY DEPTNO) where NO>=2);

4.13 Select * from EMPLOYEE where DEPTNO=20 and JOB=ANY(Select JOB from EMPLOYEE where DEPTNO=30);

4.14 Select * from EMPLOYEE where SALARY> ANY(Select MAX(SALARY) from EMPLOYEE where DEPTNO=20 or DEPTNO=30 GROUP BY DEPTNO);

4.15 Select MAX(SALARY) from EMPLOYEE GROUP BY DEPTNO HAVING MAX(SALARY) >9000;

4.16 Select MAX(SALARY) from EMPLOYEE GROUP BY ENAME HAVING MIN(SALARY)>1000 and MIN(SALARY)<5000;

JOINS

▶ Table Creation

Create table AccDept(DEPTNO number(2) PRIMARY KEY , DNAME varchar2(20), DLOC varchar2(20));

▶ Inserting values in AccDept Table

insert into AccDept values(10, 'MANAGEMENT', 'MAIN BLOCK');

insert into AccDept values(20,'DEVELOPMENT','MANUFACTURINGUNIT');

insert into AccDept values(30, 'MAINTAINANCE', 'MAIN BLOCK');

4.17 Select A.DNAME from DEPARTMENT D, ACCDEPT A where D.DEPTNO=A.DEPTNO;

4.18 Select ENAME from EMPLOYEE where DEPTNO!=ANY (Select DEPTNO from AccDept);

4.19 Select * from EMPLOYEE LEFT JOIN DEPARTMENT on DEPARTMENT.DEPTNO=EMPLOYEE.DEPTNO;

4.20 Select * from EMPLOYEE RIGHT JOIN DEPARTMENT on DEPARTMENT.DEPTNO=EMPLOYEE.DEPTNO;

4.21 Select * from EMPLOYEE FULL JOIN DEPARTMENT on DEPARTMENT.DEPTNO=EMPLOYEE.DEPTNO;

4.22 Select E.ENAME, E!.ENAME from EMPLOYEE E, EMPLOYEE E1 where E.MGR=E1.EMPNO;

4.23 Select E.ENAME, E1.SALARY from EMPLOYEE E, EMPLOYEE E1 where E.MGR=E1.EMPNO;

4.24 Select E.ENAME , E.JOB, E.EMPNO, D.DNAME, D.DLOC from EMPLOYEE E, DEPARTMENT D where E.DEPTNO=E.DEPTNO and D.DEPTNO=E.DEPTNO;

4.25 Select E.EMPNO, E.ENAME, E.JOB, E1.ENAME from EMPLOYEE E, EMPLOYEE E1 where E.MGR=E1.EMPNO;

4.26 Select E.ENAME , E1.ENAME from EMPLOYEE E, EMPLOYEE E1 where E.SALARY=E1.SALARY and E.ENAME!=E1.ENAME;

5. SET Operators and Views

5.1 Select DEPTNO from DEPARTMENT UNION Select DEPTNO from AccDept;

5.2 Select DEPTNO from DEPTUNION all Select DEPTNO from AccDept;

5.3 Select DEPTNO from DEPARTMENT INTERSECT Select DEPTNO from AccDept;

5.4 Select DEPTNO from DEPTMINUS Select DEPTNO from AccDept;

5.5 Create view manager1 as Select * from EMPLOYEE where JOB='manager';

5.6 Create view general as Select EMPNO,ENAME, EMPLOYEE.DEPTNO from EMPLOYEE, DEPARTMENT where EMPLOYEE.DEPTNO=DEPARTMENT.DEPTNO;

5.7 Create view all1 as Select EMPNO,ENAME, EMPLOYEE.DEPTNO, DNAME from EMPLOYEE, DEPARTMENT where EMPLOYEE.DEPTNO =DEPARTMENT.DEPTNO;

5.8 Select view_name from user_views;

5.9 Insert into manager values(7201,'CLAVE',20,'Computer');

5.10 Drop VIEW all1;

6. Control Structures

6.1 Write a PL/SQL program to swap two numbers without 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

```
Enter value for a: 48
Enter value for b: 11
THE PREV VALUES OF A AND B WERE
48
11
THE VALUES OF A AND B ARE
```

```
11

48
```

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;
```

<u>OUTPUT</u>

```
Enter value for a: 12

Enter value for b: 23

THE PREVIOUS VALUES OF A AND B WERE:

      12
```

```
        23
THE VALUES OF A AND B ARE:
        23
        12
```

6.3 Write a PL/SQL program to find 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

```
Enter value for a: 15
Enter value for b: 27
```

```
b IS GREATER
```

6.4 Write a PL/SQL program to find total and average of 6 subjects and display the grade

```
    declare
    daa number(10);
    dbms number(10);
    os number(10);
    rmt number(10);
    python number(10);
    cpp number(10);
    total number(10);
    per number(10);
    begin
dbms_output.put_line('ENTER THE MARKS');
dbms:=&dbms;
daa:=&daa;
cpp:=&cpp;
python:=&python;
os:=&os;
```

```
rmt:=&rmt;
total:=(daa+dbms+os+rmt+python+cpp);
per:=(total/600)*100;
if daa<40 or dbms<40 or os<40 or rmt<40 or python<40 or cpp<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

```
Enter value for dbms: 85
Enter value for daa: 74
Enter value for cpp: 71
Enter value for python: 74
Enter value for os: 76
Enter value for rmt: 70
GRADE A
PERCENTAGE IS 75
    TOTAL IS 450
```

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;
```

<u>OUTPUT</u>

```
Enter value for a: 451
sum is 10
```

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;
```

<u>OUTPUT</u>

```
Enter value for a: 765

No is:- 567
```

<u>6.7</u> 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;
```

<u>OUTPUT</u>

```
Enter value for a: 12
     11 is not a prime number
```

<u>6.8</u> Write a PL/SQL program to find factorial of 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('Factorial is'|| f);
end;
```

<u>OUTPUT</u>

```
Enter value for n: 6
Factorial is 720
```

6.9 Write a PL/SQL program to calculate the area of circle for a value of radius varying from 3 to 7

```
create table areas(radius number(10),area number(6,2));
```

PROGRAM

```
declare
pi constant number(4,2):=3.14;
radius number(5):=3;
area number(6,2);
begin
while radius<7loop
area:=pi*power(radius,2);
insert into areas values(radius,area);
radius:=radius+1;
end loop;
end;
```

OUTPUT

```
SELECT * FROM AREAS;
RADIUS AREA
---------- ----------
3 28.26
```

```
4 50.24

5 78.5

6 113.04
```

6.10 Write a PL/SQL program that will accept an account number from theuser,check if the users balance is less than minimum balance,only then deduct rs.100/- fromthe balance.this process is fired on the acct table.

- create table acct(name varchar2(10),cur_bal number(10), acctno number(6,2));

- insert into stud values('&sname',&rollno,&marks);
- 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;
```

7.Procedures and Functions

7.1 Create or replace procedure salary(deptid number) as

```
begin
Update EMPLOYEE set SALARY=SALARY+1000 where SALARY>5000 AND
DEPTNO=deptid;
end;
```

7.2 create or replace procedure salary1(empid number) as

```
 begin
update EMPLOYEE set SALARY=SALARY+SALARY*(0.1) where EMPNO=empid;
 end;
```

7.3 create or replace procedure get_sal(dept number) as

```
begin
for s in (Select * from EMPLOYEE where DEPTNO = dept)
loop
 dbms_output.put_line(s.SALARY);
 end loop;
 end;
```

<u>7.4</u> create or replace procedure get_nature(dept number) as

```
begin
for s in (Select * from EMPLOYEE where DEPTNO = dept)
 loop
dbms_output.put_line(s.job);
end loop;
end;
```

<u>7.5</u> create or replace procedure dep_name(deptid number) as

```
 begin
Select dept.dname from dept,EMPLOYEE where
EMPLOYEE.DEPTNO=dept.DEPTNO;
 end;
```

8.Triggers

8.1 Write a trigger to ensure that DEPARTMENT TABLE does not contain duplicate of null values in DEPTNO column

```
CREATE OR REPLACE TRIGGER first
BEFORE INSERT on DEPARTMENT
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 DEPARTMENT where
DEPTNO=:new.DEPTNO;
        if(a=1)  then
 raise_application_error(-20002,’error::cannot have duplicate value);
        end if;
     end if;
   END;
```

<u>8.2</u> Write a trigger to carry out the following action: on deleting a DEPTNO from DEPARTMENT table, all the records with that DEPTNO has to be deleted from the EMPLOYEE table

```
CREATE OR REPLACE TRIGGER trgr_to_delete
BEFORE DELETE ON DEPARTMENT FOR EACH ROW
DECLARE
  CURSOR get_emp( p_DEPTNO NUMBER ) IS
        Select EMPNO, ENAME, JOB, MGR, SALARY, COMM, DOB
        fromEMP
    WHERE DEPTNO=p_DEPTNO;
BEGIN
 dbms_output.put_line( 'Delete dept = ' || :old.DEPTNO );
  dbms_output.put_line( '- dept name = ' || :old.dname );
  dbms_output.put_line( '- dept loc  = ' || :old.loc );
  FOR get_emp_rec IN get_emp( :old.DEPTNO ) LOOP
    dbms_output.put( '- emp ( ' || get_emp_rec.EMPNO );
    dbms_output.put( ', ' || get_emp_rec.ename );
    dbms_output.put( ', ' || get_emp_rec.job );

dbms_output.put( ', ' || get_emp_rec.mgr );
    dbms_output.put( ', ' || get_emp_rec.SALARY );
    dbms_output.put( ', ' || get_emp_rec.comm );

dbms_output.put( ', ' || get_emp_rec.job );
   dbms_output.put_line( ' )' );
  END LOOP;
END;
```

<u>8.3</u> Write a Trigger to carry out the following action: on deleting any records from the EMPLOYEE table,the same values must be inserted into the log table.

```
CREATE TRIGGER AfterDELETETrigger on [EMP Table]
FOR DELETE
AS
INSERT INTO [logtable]( empno,[ename],[job] ,[mgr],[deptno],[sal] ,[dob])
 SELECT  empno,[ename],[job],[mgr],[deptno],[sal]  ,[dob],
 CAST( SERVERPROPERTY('MachineName') AS VARCHAR2(50))  ,
CAST( SERVERPROPERTY('ServerName') AS VARCHAR2(50))
,GETDATE()
FROM DELETED;
PRINT 'We Successfully Fired the AFTER DELETE Triggers in SQL Server and inserted a values into log Table.'
GO
```