**Practical 10**

**1)Write a PL/SQL program for calculating area of circle using FOR LOOP from table.**

SQL> declare

2 r number :=0;

3 area number :=0;

4 pi number :=3.14;

5 begin

6 for r in 0..6 loop

7 area :=pi\*r\*r;

8 insert into circle values(r,area);

9 dbms\_output.put\_line(area);

10 end loop;

11 end;

12 /

0

3.14

12.56

28.26

50.24

78.5

113.04

PL/SQL procedure successfully completed.

SQL> select \* from circle;

RADIUS AREA

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

0 0

1 3.14

2 12.56

3 28.26

4 50.24

5 78.5

1. 113.04

**2) Write a PL/SQL program for switch case displaying days of a week.**

SQL> declare

2 day char(3):='&day';

3 disp varchar2(20);

4 begin

5 disp:=

6 case day

7 when 'sun' then 'sunday'

8 when 'mon' then 'monday'

9 when 'tue' then 'tuesday'

10 when 'wed' then 'wednesday'

11 when 'thr' then 'thrusday'

12 when 'fri' then 'friday'

13 when 'sat' then 'staturday'

14 ELSE 'NO SUCH DAY'

15 END;

16 dbms\_output.put\_line('your select days is '||disp);

17 end;

18 /

Enter value for day: mon

old 2: day char(3):='&day';

new 2: day char(3):='mon';

your select days is monday

PL/SQL procedure successfully completed.

**3) Write a PL/SQL program to find maximum of 3 numbers and value inputed by user and display the same.**

SQL> declare

2 a number :='&a';

3 b number :='&b';

4 c number :='&c';

5 begin

6 dbms\_output.put\_line('a='||a||' b='||b||' c='||c);

7 if a>b AND a>c

8 then

9 dbms\_output.put\_line('a is greatest');

10 else

11 if b>a AND b>c

12 then

13 dbms\_output.put\_line('b is greatest');

14 else

15 dbms\_output.put\_line('c is greatest');

16 end if;

17 end if;

18 end;

19 /

Enter value for a: 12

old 2: a number :='&a';

new 2: a number :='12';

Enter value for b: 23

old 3: b number :='&b';

new 3: b number :='23';

Enter value for c: 26

old 4: c number :='&c';

new 4: c number :='26';

a=12 b=23 c=26

c is greatest

PL/SQL procedure successfully completed.

**4) Write a PL/SQL block calculate area of cylinder(area=2\*pi\*r\*h+2\*pi\*r\*r).**

SQL> declare

2 r number :=5;

3 pi number :=3.14;

4 h number :=10;

5 area number;

6 begin

7 area :=(2\*pi\*r\*h)+(2\*pi\*r\*r);

8 dbms\_output.put\_line('Area of Cylinder: '||area);

9 end;

10 /

Area of Cylinder: 471

PL/SQL procedure successfully completed.

**5) Write a PL/SQL block calculate area of cylinder take value from user.**

SQL> declare

2 r number :='&r';

3 pi number :=3.14;

4 h number :='&h';

5 area number;

6 begin

7 area :=(2\*pi\*r\*h)+(2\*pi\*r\*r);

8 dbms\_output.put\_line('Area of Cylinder: '||area);

9 end;

10 /

Enter value for r: 10

old 2: r number :='&r';

new 2: r number :='10';

Enter value for h: 5

old 4: h number :='&h';

new 4: h number :='5';

Area of Cylinder: 942

PL/SQL procedure successfully completed.

**6) Write a PL/SQL program for swap two number without using third variable.**

SQL> declare

2 a number :='&a';

3 b number :='&b';

4 begin

5 dbms\_output.put\_line('Before Swapping....');

6 dbms\_output.put\_line('a=' ||a||'b=' ||b);

7 dbms\_output.put\_line('After Swapping....');

8 a :=a+b;

9 b :=a-b;

10 a :=a-b;

11 dbms\_output.put\_line('a=' ||a||'b=' ||b);

12 end;

13 /

Enter value for a: 5

old 2: a number :='&a';

new 2: a number :='5';

Enter value for b: 10

old 3: b number :='&b';

new 3: b number :='10';

Before Swapping....

a=5b=10

After Swapping....

a=10b=5

PL/SQL procedure successfully completed.

**7) Write a PL/SQL program for find sum of first 5 integer.**

SQL> declare

2 s number :=0;

3 n number :=1;

4 begin

5 while n <= 5 loop

6 s := s+n;

7 n :=n+1;

8 end loop;

9 dbms\_output.put\_line('sum of first 5 integer=' ||s);

10 end;

11 /

sum of first 5 integer=15

PL/SQL procedure successfully completed.

**8) Write a PL/SQL program for find sum of 5 integer taking value from user.**

SQL> declare

2 s number :=0;

3 n number :='&n';

4 n1 number :='&n1';

5 n2 number :='&n2';

6 n3 number :='&n3';

7 n4 number :='&n4';

8 n5 number;

9 begin

10 s := s+n+n1+n2+n3+n4;

11 dbms\_output.put\_line('sum of first 5 integer=' ||s);

12 end;

13 /

Enter value for n: 1

old 3: n number :='&n';

new 3: n number :='1';

Enter value for n1: 2

old 4: n1 number :='&n1';

new 4: n1 number :='2';

Enter value for n2: 3

old 5: n2 number :='&n2';

new 5: n2 number :='3';

Enter value for n3: 4

old 6: n3 number :='&n3';

new 6: n3 number :='4';

Enter value for n4: 5

old 7: n4 number :='&n4';

new 7: n4 number :='5';

sum of first 5 integer=15

PL/SQL procedure successfully completed.

**9) Write a program for Factorial of number.**

SQL> declare

2 fact12 number :=1;

3 n number :='&n';

4 begin

5 for i in 1..n loop

6 fact12 :=fact12\*i;

7 end loop;

8 dbms\_output.put\_line('Factorial of number=' ||fact12);

9 end;

10 /

Enter value for n: 5

old 3: n number :='&n';

new 3: n number :='5';

Factorial of number=120

PL/SQL procedure successfully completed.

**Practical No. 11**

**FUNCTIONS**

1. **Write a PL/SQL function to calculate cube of number.**

SQL> create or replace function f2(a in number)

2 return number

3 IS

4 begin

5 return a\*a\*a;

6 end;

7 /

Function created.

SQL> variable n number;

SQL> begin

2 :n:=f2(3);

3 end;

4 /

PL/SQL procedure successfully completed.

SQL> print n;

N

----------

27

1. **Write a PL/SQL function maximum between two number from user.**

SQL> create or replace function f1(a in number,b in number)

2 return number

3 IS

4 begin

5 if (a>b) then

6 return a;

7 else

8 return b;

9 end if;

10 end;

11 /

Function created.

SQL> variable c number;

SQL> begin

2 :c:=f1(23,19);

3 end;

4 /

PL/SQL procedure successfully completed.

SQL> print c;

C

-------

23

1. **Write a PL/SQL function for simple interest.**

SQL> create or replace function f3(p in number,n in number,r in number)

2 return number

3 IS

4 begin

5 return ((p\*n\*r)/100);

6 end;

7 /

Function created.

SQL> variable pri number;

SQL> begin

2 :pri:=f3(10,5,6);

3 end;

4 /

PL/SQL procedure successfully completed.

SQL> print pri;

PRI

----------

3

1. **Write a PL/SQL function for entered year is Leap year or not.**

SQL> CREATE OR REPLACE FUNCTION isLeapYear(i\_year NUMBER)

2 RETURN boolean

3 AS

4 BEGIN

5 IF mod(i\_year, 400) = 0 OR ( mod(i\_year, 4) = 0 AND mod(i\_year, 100) != 0)

THEN

6 return TRUE;

7 ELSE

8 return FALSE;

9 END IF;

10 END;

11 /

Function created.

SQL> BEGIN

2 IF isLeapYear(&year) THEN

3 dbms\_output.put\_line('Yes, it is a leap year');

4 ELSE

5 dbms\_output.put\_line('No, it is not a leap year');

6 END IF;

7 END;

8 /

Enter value for year: 2000

old 2: IF isLeapYear(&year) THEN

new 2: IF isLeapYear(2000) THEN

Yes, it is a leap year

PL/SQL procedure successfully completed.

1. **Write a function to display total number of customer.**

SQL> CREATE OR REPLACE FUNCTION cu1

2 return number

3 IS

4 s number:=0;

5 BEGIN

6 select count(\*) into s from cust;

7 return s;

8 END;

9 /

Function created.

SQL> declare

2 tal number(10);

3 begin

4 tal:=cu1();

5 dbms\_output.put\_line('Total No:' ||tal);

6 end;

7 /

Total No:5

PL/SQL procedure successfully completed.

**Practical No. 12**

**PROCEDURE**

1. **Write a procedure name insert\_ user to insert data into table name user200.**

SQL> create table user200(id int,name varchar2(10));

Table created.

SQL> create or replace procedure insert\_user(id IN number,name IN varchar2)

2 is

3 begin

4 insert into user200 values(id,name);

5 end;

6 /

Procedure created.

SQL> exec insert\_user(1,'vikas');

PL/SQL procedure successfully completed.

SQL> exec insert\_user(2,'anil');

PL/SQL procedure successfully completed.

SQL> select \* from user200;

ID NAME

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

1 vikas

2 anil

SQL> begin

2 insert\_user(3,'ankul');

3 end;

4 /

PL/SQL procedure successfully completed.

SQL> select \* from user200;

ID NAME

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

1 vikas

2 anil

3 ankul

1. **Write PL/SQL procedure to print simple statement.**

SQL> create procedure hello1

2 is

3 begin

4 dbms\_output.put\_line('Hello World');

5 end;

6 /

Procedure created.

SQL> exec hello1

Hello World

PL/SQL procedure successfully completed.

1. **Write PL/SQL procedure for calculating square of number and accept the value from user.**

SQL> create or replace procedure p1(n in out number)

2 is

3 begin

4 n :=n\*n;

5 end;

6 /

Procedure created.

SQL> declare

2 n1 number:=&n1;

3 begin

4 p1(n1);

5 dbms\_output.put\_line('Square og Number: '||n1);

6 end;

7 /

Enter value for n1: 3

old 2: n1 number:=&n1;

new 2: n1 number:=3;

Square of Number: 9

PL/SQL procedure successfully completed.

1. **Write PL/SQL procedure for finding minimum of two numbers accepting from user.**

SQL> create or replace procedure p2(a in number,b in number,c out number)

2 is

3 begin

4 if a<b then

5 c:=a;

6 else

7 c:=b;

8 end if;

9 end;

10 /

Procedure created.

SQL> declare

2 n1 number:=&n1;

3 n2 number:=&n2;

4 n3 number;

5 begin

6 p2(n1,n2,n3);

7 dbms\_output.put\_line('The minimum number of two given numbers '||n1||' '||n

2||' : '||n3);

8 end;

9 /

Enter value for n1: 23

old 2: n1 number:=&n1;

new 2: n1 number:=23;

Enter value for n2: 21

old 3: n2 number:=&n2;

new 3: n2 number:=21;

The minimum number of two given numbers 23 21 : 21

PL/SQL procedure successfully completed.

1. **Write PL/SQL procedure to raise salary of employee where employee id is given.**

SQL> select \* from emp;

EMP\_ID SALARY

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

1 10000

2 20000

3 25000

4 35000

5 30000

SQL> create or replace procedure sal\_inc(emp\_id in number,sal\_raise in number)

2 as

3 begin

4 update emp set salary=salary+sal\_raise where emp\_id=emp\_id;

5 dbms\_output.put\_line('Data Updated!!!!!');

6 end;

7 /

Procedure created.

SQL> exec sal\_inc(3,5000);

Data Updated!!!!!

PL/SQL procedure successfully completed.

SQL> select \* from emp;

EMP\_ID SALARY

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

1 15000

2 25000

3 30000

4 40000

5 35000

1. **Write PL/SQL procedure for without parameter.**

SQL> create procedure hello1

2 is

3 begin

4 dbms\_output.put\_line('Hello World');

5 end;

6 /

Procedure created.

SQL> exec hello1

Hello World

PL/SQL procedure successfully completed.

1. **Write PL/SQL procedure for getting details as per employee id by user.**

SQL> create or replace procedure disp1(id number)

2 IS

3 Row employee%rowtype;

4 begin

5 select \* into row from employee where emp\_id=id ;

6 dbms\_output.put\_line('ID: '||Row.emp\_id||' '||'Name: '||Row.emp\_name||' '||

'Salary: '||Row.salary||' '||'Dept. Num.: '||Row.dept\_num);

7 End;

8 /

Procedure created.

SQL> DECLARE

2 n number:=&n;

3 BEGIN

4 disp1(n);

5 END;

6 /

Enter value for n: 1006

old 2: n number:=&n;

new 2: n number:=1006;

ID: 1006 Name: neha Salary: 25000 Dept. Num.: 201

1. **Write PL/SQL procedure for insert data into userdb table with attribute user\_id,user\_name,created\_by,created\_date using rowtype keyword.**

SQL> CREATE OR REPLACE PROCEDURE insertuserdb(

2 p\_userid IN userdb.U\_ID%TYPE,

3 p\_username IN userdb.U\_NAME%TYPE,

4 p\_createdby IN userdb.CREATED\_BY%TYPE,

5 p\_date IN userdb.CREATED\_DATE%TYPE)

6 IS

7 BEGIN

8

9 INSERT INTO userdb ("U\_ID", "U\_NAME", "CREATED\_BY", "CREATED\_DATE")

10 VALUES (p\_userid, p\_username,p\_createdby, p\_date);

11

12 DBMS\_OUTPUT.PUT\_LINE('ROW INSERTED!!!!!!!');

13

14 END;

15 /

Procedure created.

SQL> exec insertuserdb(1001,'Anil','system',SYSDATE);

ROW INSERTED!!!!!!!

PL/SQL procedure successfully completed.

SQL> select \* from userdb;

U\_ID U\_NAME CREATED\_BY CREATED\_DA

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

1001 Anil system 01-OCT-18

1002 Vikas system 01-OCT-18

**Practical no ;14**

**CURSORS**

1. **Implicit cursor :**

**Select data from table using cursor :**

SQL> DECLARE

CURSOR cust\_cur is

SELECT id, name,address,billamt FROM customer9;

cust\_rec cust\_cur%rowtype;

BEGIN

OPEN cust\_cur;

dbms\_output.put\_line('--------------------------------------------------');

dbms\_output.put\_line('| '||'ID' || ' | ' || 'NAME'||' |'||'ADDRESS'|| ' | '|| 'BILLamt'||' |');

dbms\_output.put\_line('--------------------------------------------------');

LOOP

FETCH cust\_cur into cust\_rec;

EXIT WHEN cust\_cur%notfound;

dbms\_output.put\_line('| '||cust\_rec.id || ' | ' || cust\_rec.name||'|'||cust\_rec.address || ' | ' ||

cust\_rec.billamt||' |');

END LOOP;

CLOSE cust\_cur;

END;

/

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

| ID | NAME |ADDRESS | BILLamt |

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

| 101 | vikas |mumbai | 202020 |

| 102 | vikas |mumbai | 202020 |

| 103 | vikas |mumbai | 202020 |

| 104 | vikas |mumbai | 202020 |

| 105 | vikas |mumbai | 202020 |

PL/SQL procedure successfully completed.

**Update table data using cursor :**

SQL> DECLARE

cursor c1 is select id,name from customer9;

c1\_rec c1%rowtype;

id number;

BEGIN

open c1;

fetch c1 into c1\_rec;

UPDATE customer9

SET billamt = billamt + 500;

IF c1%notfound THEN

dbms\_output.put\_line('no customers selected');

ELSIF c1%found THEN

id := sql%rowcount;

dbms\_output.put\_line( id || ' customers selected ');

END IF;

END;

/

5 customers selected

PL/SQL procedure successfully completed.

SQL> select \*from customer9;

ID NAME ADDRESS BILLAMT

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

101 vikas mumbai 203520

102 vikas mumbai 203520

103 vikas mumbai 203520

104 vikas mumbai 203520

105 vikas mumbai 203520

1. **Explicit cursor :**

**To retrieve data from table :**

SQL> DECLARE

cursor c1 is select id,name from customer9;

c1\_rec c1%rowtype;

id number;

BEGIN

open c1;

fetch c1 into c1\_rec;

UPDATE customer9

SET billamt = billamt + 500;

IF c1%notfound THEN

dbms\_output.put\_line('no customers selected');

ELSIF c1%found THEN

id := sql%rowcount;

dbms\_output.put\_line( id || ' customers selected ');

END IF;

End c1;

END;

/

5 customers selected

PL/SQL procedure successfully completed.

SQL> select \*from customer9;

ID NAME ADDRESS BILLAMT

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

101 vikas mumbai 203520

102 vikas mumbai 203520

103 vikas mumbai 203520

104 vikas mumbai 203520

105 vikas mumbai 203520

**2)retrieve data from rides table .**

DECLARE

h rides.hname%type;

r rides.racename%type;

p rides.position%type;

j rides.jockey%type;

CURSOR ride is

SELECT hname,racename,position,jockey FROM rides;

BEGIN

OPEN ride;

dbms\_output.put\_line('--------------------------------------------------');

dbms\_output.put\_line('| '||'HORSE' || ' | ' || 'RACE'||' |'||'POSITION' || ' | '|| 'JOCKEY'||' |');

dbms\_output.put\_line('--------------------------------------------------');

LOOP

FETCH ride into h,r,p,j;

EXIT WHEN ride%notfound;

dbms\_output.put\_line('| '||h|| ' | ' || r||' | '||p|| ' | ' || j||' |');

END LOOP;

CLOSE ride;

END;

/

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

| HORSE | RACE |POSITION | JOCKEY |

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

| xyz | abc | 1 | vikas |

| xyz | abc | 1 | vika |

| qwe | abc | 2 | anil |

| sed | abc | 3 | rohit |

PL/SQL procedure successfully completed.

1. **ISOPEN**

DECLARE

h rides.hname%type;

r rides.racename%type;

p rides.position%type;

j rides.jockey%type;

CURSOR vikas is

SELECT hname,racename,position,jockey FROM rides;

BEGIN

if NOT vikas%isopen then

dbms\_output.put\_line('cursor is not open ');

dbms\_output.put\_line('opening cursor ....... ');

open vikas;

else

dbms\_output.put\_line('cursor is already open');

end if;

dbms\_output.put\_line('--------------------------------------------------');

dbms\_output.put\_line('| '||'HORSE' || ' | ' || 'RACE'||' |'||'POSITION' || ' | '|| 'JOCKEY'||' |');

dbms\_output.put\_line('--------------------------------------------------');

LOOP

FETCH vikas into h,r,p,j;

EXIT WHEN vikas%notfound;

dbms\_output.put\_line('| '||h|| ' | ' || r||' | '||p|| ' | ' || j||' |');

END LOOP;

close vikas;

END;

/

cursor is not open

opening cursor .......

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

| HORSE | RACE |POSITION | JOCKEY |

| xyz | abc | 1 | vikas |

| xyz | abc | 1 | vika |

| qwe | abc | 2 | anil |

| sed | abc | 3 | rohit |

PL/SQL procedure successfully completed.

**Practical No. 15**

**Triggers**

1. **Write a tirgger to check and assign the default salary if its less than 5000.**

SQL> create table employesss(emp\_no int primary key,emp\_name varchar2(10),emp\_sa

lary int,emp\_dept\_no int,hire\_dat varchar2(10));

Table created.

SQL> create or replace trigger t1

2 before insert on employesss

3 for each row

4 begin

5 if(:new.emp\_salary <=5000) then

6 :new.emp\_salary:=5000;

7 end if;

8 end;

9 /

Trigger created.

SQL> insert into employesss values(101,'vikas',6000,201,'12-01-2018');

SQL> insert into employesss values(102,'anil',3000,202,'10-05-2018');

SQL> insert into employesss values(103,'ganesh',5000,203,'10-06-2017');

SQL> insert into employesss values(104,'rohit',1000,204,'20-06-2017');

SQL> insert into employesss values(105,'jayesh',4000,205,'20-12-2017');

SQL> select \* from employesss;

EMP\_NO EMP\_NAME EMP\_SALARY EMP\_DEPT\_NO HIRE\_DAT

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

101 vikas 6000 201 12-01-2018

102 anil 5000 202 10-05-2018

103 ganesh 5000 203 10-06-2017

104 rohit 5000 204 20-06-2017

105 jayesh 5000 205 20-12-2017

1. **Write a trigger rise an error when marks is less then 40 work on student table with attribute s\_id,s\_name, marks and percent.**

SQL> create table student(s\_id int primary key,s\_name varchar2(10),marks int,per

cen varchar2(10));

Table created.

SQL> create or replace trigger t2

2 before insert on student

3 for each row

4 begin

5 if(:new.marks <=45) then

6 Raise\_Application\_Error(-20500,'Passing is greater than 45');

7 end if;

8 end;

9 /

Trigger created.

SQL> insert into student values(01,'Ganesh',55,'55.34');

SQL> insert into student values(02,'Rohit',40,'45.24');

SQL> insert into student values(03,'Anil',49,'44.24');

SQL> select \* from student;

S\_ID S\_NAME MARKS PERCEN

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

1 Ganesh 55 55.34

2 Rohit 48 45.24

1. Anil 49 44.24
2. **Write a trigger to find username of a person performing inserting operation on the orders table with field order\_id,quantity,cost\_per\_item,total\_cost,create\_date,created\_by.**

SQL> create table orders(o\_id int primary key,quantity int,cost\_per\_item int,tot

al\_cost int,create\_date varchar2(10),created\_by varchar2(10));

Table created.

SQL> create or replace trigger orders\_before\_insert

2 before insert on orders

3 for each row

4 declare

5 v\_username varchar2(10);

6 begin

7 select user into v\_username from dual;

8 :new.create\_date:=sysdate;

9 :new.created\_by:=v\_username;

10 end;

11 /

Trigger created.

SQL> insert into orders values(001,15,100,1500,'12-10-2000','ganesh');

SQL> insert into orders values(002,20,50,1000,'12-11-2001','anil');

SQL> insert into orders values(003,10,50,500,'10-09-2005','akash');

SQL> insert into orders values(004,5,50,250,'10-10-2010','vikas');

SQL> insert into orders values(005,7,100,700,'12-10-2010','rohit');

SQL> select \* from orders;

O\_ID QUANTITY COST\_PER\_ITEM TOTAL\_COST CREATE\_DAT CREATED\_BY

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

1 15 100 1500 15-OCT-18 SYSTEM

2 20 50 1000 15-OCT-18 SYSTEM

3 10 50 500 15-OCT-18 SYSTEM

4 5 50 250 15-OCT-18 SYSTEM

5 7 100 700 15-OCT-18 SYSTEM

1. **Write a trigger to insert a value into a table from anothet table**

**Table 1: o\_id,quantity,cost\_per\_item,total\_cost (Orders Table)**

**Table 2: o\_id,quantity,cost\_per\_item,username (OrdersAudit Table)**

SQL> create table ordersss(o\_id int primary key,quantity int,cost\_per\_item int,t

otal\_cost int);

Table created.

SQL> create table orderaudit(o\_id int primary key,quantity int,cost\_per\_item int

,username varchar2(10));

Table created.

SQL> create or replace trigger t4

2 after insert on ordersss

3 for each row

4 declare

5 v\_username varchar2(10);

6 begin

7 select user into v\_username from dual;

8 insert into orderaudit(o\_id,quantity,cost\_per\_item,username) values(:new.o\_

id,:new.quantity,:new.cost\_per\_item,v\_username);

9 end;

10 /

Trigger created.

SQL> insert into ordersss values(106,10,500,5000);

SQL> insert into ordersss values(107,20,500,10000);

SQL> insert into ordersss values(108,30,100,3000);

SQL> insert into ordersss values(109,50,50,2500);

SQL> insert into ordersss values(110,100,50,5000);

SQL> select \* from orderaudit;

O\_ID QUANTITY COST\_PER\_ITEM USERNAME

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

106 10 500 SYSTEM

107 20 500 SYSTEM

108 30 100 SYSTEM

109 50 50 SYSTEM

110 100 50 SYSTEM

**Practical No. 16**

**Partitioning**

1. **Range Partition:**

SQL> create table sales(year varchar2(10),product varchar2(10),amount int)

partition by range(year)

(partition p1 values less than (1990),

partition p2 values less than (2000),

partition p3 values less than (2010),

partition p4 values less than(2020));

SQL> insert into sales values('1991','apple',80000);

SQL> insert into sales values('1999','apple',90000);

SQL> insert into sales values('2004','samsung',30000);

SQL> insert into sales values('2010','nokia',20000);

SQL> insert into sales values('2009','nokia',10000);

SQL> insert into sales values('2019','nokia',100000);

SQL> select \* from sales;

YEAR PRODUCT AMOUNT

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

1991 apple 80000

1999 apple 90000

2004 samsung 30000

2009 nokia 10000

2010 nokia 20000

2019 nokia 100000

6 rows selected.

SQL> select partition\_name,high\_value from user\_tab\_partitions where table\_name=

'SALES';

PARTITION\_NAME HIGH\_VALUE

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

P1 '1990'

P2 '2000'

P3 '2010'

P4 '2020'

SQL> select \* from sales partition(p2);

YEAR PRODUCT AMOUNT

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

1991 apple 80000

1999 apple 90000

SQL> alter table sales add partition p5 values less than(2030);

Table altered.

SQL> alter table sales rename partition p4 to p44;

Table altered.

**Split:**

SQL> alter table sales split partition p2 at (1995) into (partition p21,partitio

n p22);

Table altered.

**Merge:**

SQL> alter table sales merge partitions p21,p22 into partition p2;

Table altered.

**Truncate:**

SQL> alter table sales truncate partition p3;

Table truncated.

**Drop:**

SQL> alter table sales drop partition p3;

Table altered.

SQL> select partition\_name,high\_value from user\_tab\_partitions where table\_name=

'SALES';

PARTITION\_NAME HIGH\_VALUE

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

P1 '1990'

P2 '2000'

P44 '2020'

P5 '2030'

SQL> create table student4(roll int primary key,name varchar2(10),sem int,av\_marks int)partition by range(av\_marks)(

partition p11 values less than(45),

partition p12 values less than(75),

partition p13 values less than(100));

Table created.

SQL> insert into student4 values (1,'ankul',3,52);

SQL> insert into student4 values (2,'ankush',2,72);

SQL> insert into student4 values (3,'ankur',3,61);

SQL> insert into student4 values (4,'gaurav',5,64);

SQL> select \* from student4;

ROLL NAME SEM AV\_MARKS

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

1 ankul 3 52

2 ankush 2 72

3 ankur 3 61

1. **List Partition:-**

SQL> create table customer1(id int primary key,c\_name varchar2(10),city varchar(10),o\_place varchar2(20)) partition by list(city)

2 (partition c1 values ('north','Uk','Hp','up','Manali'),

3 partition c2 values ('south','kerla','chenai','Ap','TN'),

4 partition c3 values ('west','mumbai','pune','goa'),

5 partition c4 values ('east','WB','Asam','agartala'));

SQL> insert into customer1 values(1,'sujit','Hp','laptop');

SQL> insert into customer1 values(2,'Adarsh','mumbai','cellphone');

SQL> insert into customer1 values(3,'Ankul','kerla','Idli');

SQL> insert into customer1 values(4,'anshuman','WB','PD');

SQL> select \* from customer1;

ID C\_NAME CITY O\_PLACE

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

1 sujit Hp laptop

3 Ankul kerla Idli

2 Adarsh mumbai cellphone

4 anshuman WB PD

SQL> alter table customer1 add partition c5 values(default);

Table altered.

SQL> insert into customer1 values(5,'Adi','odisa','camera');

SQL> select \* from customer1 partition(c5);

ID C\_NAME CITY O\_PLACE

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

5 Adi odisa camera

SQL> alter table customer1 split partition c4 values('east')into (partition c6,p

artition c7);

Table altered.

SQL> alter table customer1 modify partition c2 add values('ts');

Table altered.

SQL> alter table customer1 merge partitions c1,c2 into partition c23;

SQL> alter table customer 1 drop partition (c6);

Table altered.

1. **Hash Partition:**

create table product(p\_no number primary key,description varchar(20))

partition by hash(p\_no) partitions 4;

Table created.

SQL> insert into product values(1,'pen');

SQL> insert into product values(2,'pen');

SQL> insert into product values(3,'pencil');

SQL> insert into product values(4,'fan');

SQL> insert into product values(5,'bottle');

SQL> insert into product values(6,'mouse');

SQL> insert into product values(7,'notebook');

SQL> insert into product values(8,'keyboard');

SQL> insert into product values(9,'PC');

SQL> insert into product values(10,'charger');

SQL> select \*from product;

PNO DESCRIPTION

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

6 mouse

9 PC

10 charger

2 pen

5 bottle

8 keyboard

1 pen

3 pencil

4 fan

7 notebook

10 rows selected.

SQL> select partition\_name,high\_value from user\_tab\_partitions where table\_name=

'PRODUCT3';

PARTITION\_NAME HIGH\_VALUE

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

SYS\_P21

SYS\_P22

SYS\_P23

SYS\_P24

SQL> select \*from product partition(SYS\_P24);

PNO DESCRIPTION

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

1 pen

3 pencil

4 fan

7 notebook

SQL> alter table product add partition;

Table altered.

SQL> select partition\_name,high\_value from user\_tab\_partitions where table\_name=

'PRODUCT';

PARTITION\_NAME HIGH\_VALUE

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

SYS\_P21

SYS\_P22

SYS\_P23

SYS\_P24

SYS\_P25

SQL> alter table product rename partition SYS\_P21 to jayesh;

Table altered.

SQL> select partition\_name,high\_value from user\_tab\_partitions where table\_name=

'PRODUCT';

PARTITION\_NAME HIGH\_VALUE

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

JAYESH

SYS\_P22

SYS\_P23

SYS\_P24

SYS\_P25

SQL> alter table product truncate partition jayesh;

Table truncated.

SQL> select partition\_name,high\_value from user\_tab\_partitions where table\_name=

'PRODUCT';

PARTITION\_NAME HIGH\_VALUE

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

JAYESH

SYS\_P22

SYS\_P23

SYS\_P24

SYS\_P25

SQL> select \*from product partition(jayesh);

no rows selected

SQL> select \*from product partition(SYS\_P24);

PNO DESCRIPTION

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

1 pen

3 pencil

4 fan

7 notebook

SQL> alter table product drop partition jayesh;

alter table product drop partition jayesh

\*

ERROR at line 1:

ORA-14255: table is not partitioned by Range, List, Composite Range or

Composite List method

SQL> alter table product drop partition (jayesh);

alter table product drop partition (jayesh)

\*

ERROR at line 1:

ORA-14255: table is not partitioned by Range, List, Composite Range or

Composite List method

**TABLESPACE**

SQL> create tablespace tbs3 Datafile 'tbs3.dat' size 20M online;

Tablespace created.

SQL> create table sales\_hash (s\_id int,s\_date varchar2(20),c\_id int,price int) p

artition by hash(s\_id)

2 (partition p1 tablespace tbs1,

3 partition p2 tablespace tbs2,

4 partition p3 tablespace tbs3);

Table created.

SQL> insert into sales\_hash values(1,'10-jan-2010',1001,10100);

SQL> insert into sales\_hash values(2,'11-mar-2012',1002,1000);

SQL> insert into sales\_hash values(3,'20-Feb-1997',1003,2000);

SQL> insert into sales\_hash values(4,'20-Aug-2010',1004,2050);

SQL> insert into sales\_hash values(5,'01-oct-2015',1005,5000);

SQL> select \* from sales\_hash;

S\_ID S\_DATE C\_ID PRICE

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

1 10-jan-2010 1001 10100

3 20-Feb-1997 1003 2000

4 20-Aug-2010 1004 2050

2 11-mar-2012 1002 1000

5 01-oct-2015 1005 5000

SQL> select \* from sales\_hash partition(p1);

no rows selected

SQL> select \* from sales\_hash partition(p2);

S\_ID S\_DATE C\_ID PRICE

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

1 10-jan-2010 1001 10100

3 20-Feb-1997 1003 2000

4 20-Aug-2010 1004 2050

SQL> select \* from sales\_hash partition(p3);

S\_ID S\_DATE C\_ID PRICE

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

2 11-mar-2012 1002 1000

5 01-oct-2015 1005 5000

**Experiment No: 17**

**Fragmentation**

1. SQL> create table employes(e\_no int primary key,e\_name varchar2(20),e\_add

varchar2(20),e\_salary int,e\_desig varchar2(10));

Table created.

SQL> insert into employes values(1001,'viki','powai',25000,'MD');

SQL> insert into employes values(1002,'jaspreet','powai',25000,'SD');

SQL> insert into employes values(1003,'jagjit','vikhroli',24000,'programer');

SQL> insert into employes values(1004,'jaspal','Aroli',30000,'contractor');

SQL> insert into employes values(1005,'satish','thane',28000,'propriator');

SQL> insert into employes values(1006,'Anil','ghatkopar',20000,'clerck');

SQL> insert into employes values(1007,'Vivek','Mira-road',22120,'mover');

SQL> insert into employes values(1008,'MAni','Mahim',5422,'Accountant');

SQL> insert into employes values(1009,'Rohit','ghansholi',65480,'CEO');

SQL> insert into employes values(1010,'priya','Nerul',32000,'DBA');

SQL> insert into employes values(1011,'Gautami','chinchpokli',15000,'teacher');

SQL> insert into employes values(1012,'joan','chinchpokli',18000,'teacher');

SQL> select \* from employes;

E\_NO E\_NAME E\_ADD E\_SALARY E\_DESIG

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

1001 viki powai 25000 MD

1002 jaspreet powai 25000 SD

1003 jagjit vikhroli 24000 programer

1004 jaspal Aroli 30000 contractor

1005 satish thane 28000 propriator

1006 Anil ghatkopar 20000 clerck

1007 Vivek Mira-road 22120 mover

1008 MAni Mahim 5422 Accountant

1009 Rohit ghansholi 65480 CEO

1010 priya Nerul 32000 DBA

1011 Gautami chinchpokli 15000 teacher

1012 joan chinchpokli 18000 teacher

SQL> create table employes1 as(select \* from employes where e\_salary<10000);

SQL> create table employes2 as(select \* from employes where e\_salary between 10000 and

20000);

SQL> create table employes3 as(select \* from employes where e\_salary between 10000 and

20000);

Table created.

SQL> select e\_salary from employes;

E\_SALARY

----------

25000

25000

24000

30000

28000

20000

22120

5422

65480

32000

15000

18000

SQL> select \* from employes where e\_salary=20000;

E\_NO E\_NAME E\_ADD E\_SALARY E\_DESIG

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

1006 Anil ghatkopar 20000 clerck

SQL> select e\_name,e\_salary from employes where e\_name like 'j%';

E\_NAME E\_SALARY

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

jaspreet 25000

jagjit 24000

jaspal 30000

SQL> select e\_salary from employes1;

E\_SALARY

----------

5422

SQL> select \* from employes2 where e\_salary=20000;

E\_NO E\_NAME E\_ADD E\_SALARY E\_DESIG

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

1006 Anil ghatkopar 20000 clerck

SQL> select e\_name,e\_salary from employes3 where e\_name like 'j%';

E\_NAME E\_SALARY

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

joan 18000

SQL> create table employess(e\_no int NULL,e\_name varchar2(20) NULL,e\_add

varchar2(20),e\_mail varchar2(20),e\_salary int);

Table created.

SQL> insert into employes values(1001,'viki','powai','viki@gmail.com',25000);

SQL> insert into employess values(1001,'viki','powai','viki@gmail.com',25000);

SQL> insert into employess values(1002,'anil','ghatkopar','anil@gmail.com',10000);

SQL> insert into employess values(1003,'rohit','aroli','ad@gmail.com',12500);

SQL> insert into employess values(1003,'','mira-road','joshi@gmail.com',21200);

SQL> insert into employess values(1004,'Vivek','mira-road','vivek@gmail.com',25000);

SQL> insert into employess values(1005,'satish','kalyan','yadav@gmail.com',28500);

SQL> insert into employess values(1006,'','titwala','tl@gmail.com',30100);

SQL> insert into employess values(1007,'sachin','titwala','sachin@gmail.com',30100);

SQL> insert into employess values(1008,'','dadar','Hm@gmail.com',25632);

SQL> insert into employess values(1009,'patil;','virar','patil@gmail.com',26522);

SQL> select \* from employess;

E\_NO E\_NAME E\_ADD E\_MAIL E\_SALARY

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

1001 viki powai viki@gmail.com 25000

1002 anil ghatkopar anil@gmail.com 10000

1003 rohit aroli ad@gmail.com 12500

1003 mira-road joshi@gmail.com 21200

1004 Vivek mira-road vivek@gmail.com 25000

1005 satish kalyan yadav@gmail.com 28500

1006 titwala tl@gmail.com 30100

1007 sachin titwala sachin@gmail.com 30100

1008 dadar Hm@gmail.com 25632

1009 patil virar patil@gmail.com 26522

SQL> create table employess1 as(select e\_no,e\_name,e\_add from employess);

SQL> create table employess2 as(select e\_no,e\_mail,e\_salary from employess);

Table created.

SQL> select e\_salary from employess2 where e\_no=1005;

E\_SALARY

----------

28500

SQL> select e2.e\_mail from employess1 e1,employess2 e2 where e1.e\_no=e2.e\_no;

E\_MAIL

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

viki@gmail.com

anil@gmail.com

ad@gmail.com

ad@gmail.com

joshi@gmail.com

joshi@gmail.com

vivek@gmail.com

yadav@gmail.com

tl@gmail.com

sachin@gmail.com

Hm@gmail.com

[patil@gmail.com](mailto:patil@gmail.com)

SQL> select e1.e\_name from employess1 e1,employess2 e2 where e1.e\_no=e2.e\_no and

e2.e\_salary > 20000;

E\_NAME

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

viki

rohit

Vivek

satish

sachin

patil;

SQL> select e1.e\_name,e2.e\_mail from employess1 e1,employess2 e2 where e1.e\_no=e

2.e\_no and e1.e\_no is not null;

E\_NAME E\_MAIL

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

viki viki@gmail.com

anil anil@gmail.com

rohit ad@gmail.com

rohit joshi@gmail.com

Vivek vivek@gmail.com

satish yadav@gmail.com

sachin sachin@gmail.com

patil; [patil@gmail.com](mailto:patil@gmail.com)

**Practical no.18**

**ADT**

**(Abstract data type)**

**1.describe all.**

SQL> desc address;

Name Null? Type

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

STREET NUMBER(38)

CITY VARCHAR2(20)

STATE VARCHAR2(20)

ZIP NUMBER(38)

SQL> desc person3;

Name Null? Type

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

CID NOT NULL NUMBER(38)

ADDDD ADDRESS

**2.insert 5 records.**

Inserted 5 records.

**3) find all the customers with city starts from m or likewise.**

SQL> select c.cid,c.adddd from person3 c where c.adddd.city like 'm%';

CID ADDDD(STREET, CITY, STATE, ZIP)

101 ADDRESS(111, 'mumbai', 'maharashtra', 333)

102 ADDRESS(112, 'mumbai', 'maharashtra', 444)

**4)update customers state.**

SQL> update person3 p set p.adddd.state='bihar' where p.cid=101;

1 row updated.

SQL> select \*from person3;

CID ADDDD(STREET, CITY, STATE, ZIP)

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

101 ADDRESS(111, 'mumbai', 'bihar', 333)

102 ADDRESS(112, 'mumbai', 'maharashtra', 444)

103 ADDRESS(113, 'thane', 'maharashtra', 555)

104 ADDRESS(114, 'ghodbandar', 'maharashtra', 666)

105 ADDRESS(115, 'vadala', 'maharashtra', 777)

**5)delete customer records using zipcode.**

SQL> delete person3 p where p.adddd.zip=444;

1 row deleted.

SQL> select \*from person3;

CID ADDDD(STREET, CITY, STATE, ZIP)

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

101 ADDRESS(111, 'mumbai', 'bihar', 333)

103 ADDRESS(113, 'thane', 'maharashtra', 555)

104 ADDRESS(114, 'ghodbandar', 'maharashtra', 666)

105 ADDRESS(115, 'vadala', 'maharashtra', 777)

**Reference and dereference**

**1)create a type person(pname,phno)**

SQL> create type per as object

2 (

3 pname varchar(20),

4 phone int

5 );

6 /

Type created.

**2)create table of above type**

SQL> create table person5 of per;

Table created.

SQL> desc person5;

Name Null? Type

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

PNAME VARCHAR2(20)

PHONE NUMBER(38)

**3)insert 5 values**

SQL> insert into person5 values(per('vikas',8888110416));

SQL> insert into person5 values(per('anil',9604270262));

SQL> insert into person5 values(per('akshay',9999999999));

SQL> insert into person5 values(per('ganesh',9999999999));

SQL> insert into person5 values(per('rohit',9999999999));

**4)display all from table**

**SQL> select \* from person5;**

PNAME PHONE

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

vikas 8888110416

anil 9604270262

akshay 9999999999

ganesh 9999999999

rohit 9999999999

**5)create table contact with column name as number ref by type person**

SQL> create table contact

2 (

3 cnum ref per,

4 cdate date

5 );

Table created.

SQL> desc contact;

Name Null? Type

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

CNUM REF OF PER

CDATE DATE

**6)insert values in new table**

SQL> insert into contact values((select ref(p) from person5 p where p.pname ='vikas'),'17-march-2018');

1 row created.

SQL> select \* from contact;

CNUM CDATE

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

000022020865A5030694D3983ADA3BC656332F8F66B41AD780D4FC6B04CF02E684B0ED9 17-MAR-18

**7)display all the value from new table**

SQL> select \* from contact;

CNUM CDATE

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

000022020865A503028B694D3983ADA3BC656332F8F66B41AD780D4FC6B04CF02 17-MAR-18

000022020865A503028B694D3983ADA3BC656332F8F66B41AD780D4FC6B04CF02 17-MAR-18

**8)apply dereference**

**SQL> select deref(cnum),cdate from contact;**

DEREF(CNUM)(PNAME, PHONE) CDATE

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

PER('vikas', 8888110416) 17-MAR-18

PER('vikas', 8888110416) 17-MAR-18

**ADT Inheritance**

**1)create types**

SQL> create type person8 as object(ssn int, pname varchar(20),address varchar(20)) not final;

2 /

Type created.

SQL> create type std8 under person8(dept varchar(20),major varchar(20)) not final;

2 /

Type created.

SQL> create type emp8 under person8(eid int,manager varchar(20)) final;

2 /

Type created.

SQL> create type parttimestd under std8(noofhrs int) final;

2 /

Type created.

SQL> create table person9 of person8;

Table created.

SQL>

Table created.

**2) insert data into tables :**

SQL> insert into person9 values(person8(11112,'vikas','thane'));

1 row created.

SQL> insert into person9 values(std8(11113,'anil','mumbai','mca','xyz'));

1 row created.

SQL> insert into person9 values(emp8(11113,'anil','mumbai',003,'akshay'));

1 row created.

SQL> insert into person9 values(parttimestd(11114,'rohit','kurla','mms','abc',3));

1 row created.

**3)select data :**

SQL> SELECT \* FROM person9 p;

SSN PNAME ADDRESS

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

11112 vikas thane

11113 anil mumbai

11113 anil mumbai

11114 rohit kurla

**SQL> select value(p) from person9 p where value(p) is of (std8);**

VALUE(P)(SSN, PNAME, ADDRESS)

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

STD8(11113, 'anil', 'mumbai', 'mca', 'xyz')

PARTTIMESTD(11114, 'rohit', 'kurla', 'mms', 'abc', 3)

SQL> select value(p) from person9 p where value(p) is of (emp8);

VALUE(P)(SSN, PNAME, ADDRESS)

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

EMP8(11113, 'anil', 'mumbai', 3, 'akshay')

SQL> select value(p) from person9 p where value(p) is of (parttimestd);

VALUE(P)(SSN, PNAME, ADDRESS)

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

PARTTIMESTD(11114, 'rohit', 'kurla', 'mms', 'abc', 3)