DEPARTMENT OF COMPUTER SCIENCE AND IT

RDBMS LAB

MANUAL

PREPARED BY:

S.PRABHAVATHI M.SC. M.PHIL, B.ED,
DEPARTMENT OF COMPUTER SCIENCE AND IT
JAMAL MOHAMED COLLEGE
TRICHY -20

RDBMS LAB

Course Code: 23PCS2CC8P1 **Class : I M.Sc.**

CONTENTS

DATA DEFINITION LANGUAGE

Ex No. 1.1 Table Creation – Customer, Account, Loan, Branch, Depositor, Borrower, Supplier

//CUSTOMER TABLE

SQL> create table cust1(cid number(5) primary key,cname varchar(10),address varchar(10));

Table created.

SQL> desc cust1;

Name	Null?	Type
CID	NOT NULL	NUMBER(5)
CNAME		VARCHAR2(10)
ADDRESS		VARCHAR2(10)

//ACCOUNT TABLE

SQL> create table act(ano number(10) primary key,bname varchar(10),balance number(10));

Table created.

SQL> desc act;

Name	Null?	Type
ANO	NOT NULL	NUMBER(10)
BNAME		VARCHAR2(10)
BALANCE		NUMBER(10)

//LOAN TABLE

SQL> create table l1(lno number(10) primary key,bname varchar(10),amount number(10));

Table created.

SQL> desc l1;

Name	Null?	Type
LNO	NOT NULL	NUMBER(10)
BNAME		VARCHAR2(10)
AMOUNT		NUMBER(10)

//BRANCH TABLE

SQL> create table br(bname varchar(10),bcity varchar(10),asset number(10));

Table created.

SQL> desc br;

Name	Null?	Type
BNAME		VARCHAR2(10)
BCITY		VARCHAR2(10)
ASSET		NUMBER(10)

//DEPOSITER TABLE

SQL> desc dep;

Name	Null?	Type
CNAME		VARCHAR2(10)
ANO		NUMBER(15)

//BORROWER TABLE

SQL> create table bl(cname varchar(10),lno number(10),foreign key(lno)references l1(lno)on delete cascade);

Table created.

SQL> desc bl;

Name	Null?	Type
CNAME		VARCHAR2(10)
LNO		NUMBER(10)

//SUPPLIER TABLE

SQL> create table sp(sno number(5),partno number(5),color varchar(8),quality number(10));

Table created.

SQL> desc sp;

Name	Null?	Type
SNO		NUMBER(5)
PARTNO		NUMBER(5)
COLOR		VARCHAR2(8)
QUALITY		NUMBER(10)

Ex No. 1.2 To perform foreign key on delete cascade

//LOAN TABLE

SQL> create table ln1(lno number(10)primary key,bname varchar(10),amount number(10));

Table created.

SQL> desc ln1;

Name	Null?	Type
LNO	NOT NULL	NUMBER(10)
BNAME		VARCHAR2(10)
AMOUNT		NUMBER(10)

```
SQL> insert into ln1 values(&lno,'&bname',&amount);
Enter value for lno: 1001
Enter value for bname: kk nagar
Enter value for amount: 4000
old   1: insert into ln1 values(&lno,'&bname',&amount)
new   1: insert into ln1 values(1001,'kk nagar',4000)
1 row created.
```

SQL> select*from ln1;

LNO	BNAME	AMOUNT
1001	kk nagar	4000
1002	airport	4001
1003	jk nagar	4003
1004	skyland	4004

//BORROWER TABLE

SQL> create table bo1(cname varchar(10),lno number(10),foreign key(lno)references ln1(lno)on delete cascade);

Table created.
SQL> insert into bo1 values('&cname',&lno);
Enter value for cname: nami
Enter value for lno: 1001
old 1: insert into bo1 values('&cname',&lno)
new 1: insert into bo1 values('nami',1001)
1 row created.

SQL> select*from bo1;

CNAME LNO

---

nami 1001

luffy 1002

sanji 1003

zoro 1004
SQL> delete from ln1 where lno='1001';
1 row deleted.
SQL> select*from ln1;

LNO BNAME AMOUNT

1002 airport 4001

1003 jk nagar 4003

1004 skyland 4004
SQL> delete from bo1 where lno='1001';
1 rows deleted.
SQL> select*from bo1;
CNAME LNO

---

luffy 1002

sanji 1003

zoro 1004
SQL> commit;
Commit complete.

Ex No. 1.3 Alter – Add columns, Add constraints, Modify (datatype & size), Drop table.

//ADD COLUMN

SQL> desc dep;

Name	Null?	Type
CNAME		VARCHAR2(10)
ANO		NUMBER(15)

SQL> alter table dep add city varchar(20);

Table altered.

//ADD CONSTRAINTS

SQL> desc dep;

Name	Null?	Type
CNAME		VARCHAR2(10)
ANO		NUMBER(15)
CITY		VARCHAR2(20)

SQL> alter table dep add primary key(ano);

Table altered.

SQL> desc dep;

Name	Null?	Type
CNAME		VARCHAR2(10)
ANO	NOT NULL	NUMBER(15)
CITY		VARCHAR2(20)

//MODIFY

SQL> alter table act modify ano number(20);

Table altered.

SQL> desc act;

Name	Null?	Type
ANO	NOT NULL	NUMBER(20)
BNAME		VARCHAR2(10)
BALANCE		NUMBER(10)

//RENAME

SQL> alter table act rename column balance to amount;

Table altered.

SQL> desc act;

Name	Null?	Type
ANO	NOT NULL	NUMBER(20)
BNAME		VARCHAR2(10)
AMOUNT		NUMBER(10)

//DELETE

SQL> alter table dep drop column city;

Table altered.

SQL> desc dep;

Name	Null?	Type
CNAME		VARCHAR2(10)
ANO	NOT NULL	NUMBER(15)

SQL> commit;

Commit complete.

DATA MANIPULATION LANGUAGE

Ex No. 2.1 Insertion

//CUSTOMER TABLE

```
SQL> desc cust1;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
CID                                       NOT NULL NUMBER(5)
CNAME                                              VARCHAR2(10)
ADDRESS                                            VARCHAR2(10)
SQL> insert into cust1 values(&cid,'&cname','&address');
Enter value for cid: 101
Enter value for cname: robin
Enter value for address: japan
old   1: insert into cust1 values(&cid,'&cname','&address')
new   1: insert into cust1 values(101,'robin','japan')
1 row created.
SQL> select*from cust1;
CID      CNAME    ADDRESS
---------- ---------- ----------
101        robin      japan
102        nami       skyland
103        vivi       dessert
104        ragu       trichy
```

//ACCOUNT TABLE

SQL> desc act;

Name	Null?	Type
ANO	NOT NULL	NUMBER(20)
BNAME		VARCHAR2(10)
AMOUNT		NUMBER(10)

SQL> insert into act values(&ano,'&bname',&balance);
Enter value for ano: 12345
Enter value for bname: cantonment
Enter value for balance: 20000
old 1: insert into act values(&ano,'&bname',&balance)
new 1: insert into act values(12345,'cantonment',20000)
1 row created.
SQL> select*from act;

ANO	BNAME	AMOUNT
12345	cantonment	20000
12346	trichy	19000
12347	jk nagar	17000
12348	kk nagar	15000

//LOAN TABLE

SQL> desc l1;

Name	Null?	Type
LNO	NOT NULL	NUMBER(10)
BNAME		VARCHAR2(10)
AMOUNT		NUMBER(10)

SQL> insert into l1 values(&lno,'&banme',&amount);
Enter value for lno: 1001
Enter value for banme: cantonment
Enter value for amount: 80000
old 1: insert into l1 values(&lno,'&banme',&amount)
new 1: insert into l1 values(1001,'cantonment',80000) 1
row created.

SQL> select*from l1;

LNO	BNAME	AMOUNT
1001	cantonment	80000
1002	rockfort	70000
1003	bazar	50000
1004	simco	40000

//BRANCH TABLE

SQL> desc br;

Name	Null?	Type
BNAME		VARCHAR2(10)
BCITY		VARCHAR2(10)
ASSET		NUMBER(10)

SQL> insert into br values('&bname','&bcity',&asset);
Enter value for bname: cantonment
Enter value for bcity: trichy
Enter value for asset: 50000
old 1: insert into br values('&bname','&bcity',&asset)
new 1: insert into br values('cantonment','trichy',50000)
1 row created.

SQL> select*from br;

BNAME	BCITY	ASSET
cantonment	trichy	50000
rockfort	madurai	40000
t nagar	chennai	30000
kk nagar	salem	60000

//DEPOSITER TABLE

```
SQL> desc dep;

Name                                      Null?    Type
----------------------------------------- -------- ----------------------------

CNAME                                              VARCHAR2(10)

ANO                                       NOT NULL NUMBER(15)
SQL> insert into dep values('&cname',&ano);
Enter value for cname: thasu
Enter value for ano: 12345
old   1: insert into dep values('&cname',&ano)
new   1: insert into dep values('thasu',12345)
1 row created.
SQL> select*from dep;

CNAME      ANO
---------- ----------
thasu      12345
shafu      12346
ragu       12347
priya      12348
```

//BORROWER TABLE

```
SQL> desc bl;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------

CNAME                                              VARCHAR2(10)

LNO                                                NUMBER(10)
SQL> insert into bl values('&cname',&lno);
Enter value for cname: ragu
Enter value for lno: 1001
old   1: insert into bl values('&cname',&lno)
new   1: insert into bl values('ragu',1001)
1 row created.
```

SQL> select*from bl;

CNAME	LNO
ragu	1001
thasu	1002
luffy	1003
zoro	1004

//SUPPLIER TABLE

SQL> desc sp;

Name	Null?	Type
SNO		NUMBER(5)
PARTNO		NUMBER(5)
COLOR		VARCHAR2(8)
QUALITY		NUMBER(10)

```
SQL> insert into sp values(&sno,&partno,'&color',&quality);
Enter value for sno: 1
Enter value for partno: 101
Enter value for color: red
Enter value for quality: 7
old   1: insert into sp values(&sno,&partno,'&color',&quality)
new   1: insert into sp values(1,101,'red',7)
1 row created.
```

SQL> select*from sp;

SNO	PARTNO	COLOR	QUALITY
1	101	red	7
2	102	green	5
3	103	yellow	3
4	104	purple	9

```
SQL> commit;
Commit complete.
```

Ex No. 2.2 Arithmetic, Logical, Comparison operations

ARITHMETIC OPERATORS

SQL> create table empp2(eno number(5),ename varchar(10),salary number(10));

Table created.

SQL> desc empp2;

Name	Null?	Type
ENO		NUMBER(5)
ENAME		VARCHAR2(10)
SALARY		NUMBER(10)

SQL> insert into empp2 values(&eno,'&ename',&salary);

Enter value for eno: 2201

Enter value for ename: ragu

Enter value for salary: 500000

old 1: insert into empp2 values(&eno,'&ename',&salary)

new 1: insert into empp2 values(2201,'ragu',500000)

1 row created.

SQL> select*from empp2;

ENO	ENAME	SALARY
2201	ragu	500000
2202	thasu	700000
2203	jack	400000
2204	shajee	600000

//ADDITION

SQL> select eno,ename,salary+100as"bonus"from empp2;

ENO	ENAME	bonus
2201	ragu	500100
2202	thasu	700100
2203	jack	400100
2204	shajee	600100

//SUBTRACTION

SQL> select eno,ename,salary-100as"decrement"from empp2;

ENO	ENAME	decrement
2201	ragu	499900
2202	thasu	699900
2203	jack	399900
2204	shajee	599900

//MULTIPLICATION

SQL> select eno,ename,salary*100as"increment"from empp2;

ENO	ENAME	increment
2201	ragu	50000000
2202	thasu	70000000
2203	jack	40000000
2204	shajee	60000000

//DIVISION

```
SQL> select  eno,ename,salary/100as"incentive"from empp2;

ENO      ENAME    incentive
-------                 ----------
2201     ragu            5000
2202     thasu           7000
2203     jack            4000
2204     shajee          6000
```

LOGICAL OPERATOR

```
SQL> select*from empp2;

ENO       ENAME     SALARY
----------  ----------  ----------
2201      ragu            500000
2202      thasu           700000
2203      jack            400000
2204      shajee          600000
SQL> create table empp3(eno number(5),ename varchar(10),salary number(20),
        age number(5));
Table created.
SQL> desc empp3;
Name                                      Null?    Type
----------------------------------------- -------- ----------------------------
ENO                                                NUMBER(5)
ENAME                                              VARCHAR2(10)
SALARY                                             NUMBER(20)
AGE                                                NUMBER(5)
SQL> insert into empp3 values(&eno,'&ename',&salary,&age);
Enter value for eno: 2201
Enter value for ename: ragu
Enter value for salary: 500000
Enter value for age: 20
old      1: insert into empp3 values(&eno,'&ename',&salary,&age)
new      1: insert into empp3 values(2201,'ragu',500000,20)
1 row created.
```

SQL> select*from empp3;

ENO	ENAME	SALARY	AGE
2201	ragu	500000	20
2202	thasu	700000	21
2203	jack	400000	19
2204	shajee	300000	18

//AND

SQL> select eno,ename,salary,age from empp3 where salary<400000 AND age<19;

ENO	ENAME	SALARY	AGE
2204	shajee	300000	18

//OR

SQL> select eno,ename,salary,age from empp3 where salary>300000 OR age>18;

ENO	ENAME	SALARY	AGE
2201	ragu	500000	20
2202	thasu	700000	21
2203	jack	400000	19

//NOT

SQL> select eno,ename,salary,age from empp3 where NOT age=18;

ENO	ENAME	SALARY	AGE
2201	ragu	500000	20
2202	thasu	700000	21
2203	jack	400000	19

COMPARISON OPERATOR

SQL> select*from act;

ANO	BNAME	AMOUNT
12345	cantonment	20000
12346	trichy	19000
12347	jk nagar	17000
12348	kk nagar	15000

//EQUAL TO (=)

SQL> select*from act where amount=15000;

ANO	BNAME	AMOUNT
12348	kk nagar	15000

//GREATER THAN (>)

SQL> select*from act where amount>19000;

ANO	BNAME	AMOUNT
12345	cantonment	20000

//LESS THAN (<)

SQL> select*from act where amount<20000;

ANO	BNAME	AMOUNT
12346	trichy	19000
12347	jk nagar	17000
12348	kk nagar	15000

//GREATER THAN OR EQUAL TO (>=)

SQL> select*from act where amount>=17000;

ANO	BNAME	AMOUNT
12345	cantonment	20000
12346	trichy	19000
12347	jk nagar	17000

//LESS THAN OR EQUAL TO (<=)

SQL> select*from act where amount<=17000;

ANO	BNAME	AMOUNT
12347	jk nagar	17000
12348	kk nagar	15000

//NOT EQUAL TO (<>)

SQL> select*from act where amount<>20000;

ANO	BNAME	AMOUNT
12346	trichy	19000
12347	jk nagar	17000
12348	kk nagar	15000

SQL> commit

Commit complete.

Ex No. 2.3 String Operations

SQL> select*from cust1;

CID	CNAME	ADDRESS
101	robin	japan
102	nami	skyland
103	vivi	dessert
104	ragu	trichy

//FIND THE CUSTOMER NAMES WHOSE NAMES START WITH R:

SQL> select cname from cust1 where cname like'r%';

CNAME

---

robin

ragu

//FIND THE CUSTOMER NAMES WHOSE NAMES END WITH I:

SQL> select cname from cust1 where cname like'%i';

CNAME

---

nami

vivi

//FIND THE CUSTOMER NAME WHOSE NAMES CONTAIN _ 'am' || AS A SUBSTRING:

SQL> select cname from cust1 where cname like'%am%';

CNAME

---

nami

//FIND THE CUSTOMER NAME WHOSE NAMES EXACTLY FIVE CHARACTER:

SQL> select cname from cust1 where cname like'_____';

CNAME

Robin

//FIND THE CUSTOMER NAME WHOSE NAMES ATLEAST FOUR CHARACTER:

SQL> select cname from cust1 where cname like'____%';

CNAME

----

robin

nami

vivi

ragu

SQL> commit;

Commit complete.

Ex No. 2.4 Tuple Variables

```
SQL> select*from br;

BNAME      BCITY      ASSET
---------- ---------- ----------
cantonment trichy     50000
rockfort   madurai    40000
t nagar    chennai    30000
kk nagar   chennai    60000

SQL> select t.bname from br t,br s where t.asset>s.asset and s.bcity='chennai';

BNAME
----------
kk nagar

SQL> commit;
Commit complete.
```

Ex No. 2.5 Ordering of Tuples

```
SQL> select*from cust1;

CID        CNAME      ADDRESS
---------- ---------- ----------
101        robin      japan
102        nami       skyland
103        vivi       dessert
104        ragu       trichy

SQL> select cname from cust1 order by cname asc;

CNAME
----------
nami
ragu
robin
vivi

SQL> select cname from cust1 order by cname desc;

CNAME
----------
vivi
robin
ragu
nami

SQL> commit;

Commit complete.
```

Ex No. 2.6 Set Operation

SQL> select*from ln1;

LNO	BNAME	AMOUNT
1002	airport	4001
1003	jk nagar	4003
1004	skyland	4004

SQL> select*from act;

ANO	BNAME	AMOUNT
12345	cantonment	20000
12346	trichy	19000
12347	jk nagar	17000
12348	kk nagar	15000

//UNION ALL

SQL> select bname from ln1 union all select bname from act;

BNAME

---

airport

jk nagar

skyland

cantonment

trichy

jk nagar

kk nagar

7 rows selected.

//UNION

SQL> select bname from ln1 union select bname from act;

BNAME

---

airport

cantonment

jk nagar

kk nagar

skyland

trichy

6 rows selected.

//INTERSECT

SQL> select bname from ln1 intersect select bname from act;

BNAME

---

jk nagar

//MINUS

SQL> select bname from ln1 minus select bname from act;

BNAME

---

airport

skyland

SQL> commit;

Commit complete.

Aggregate Functions

Ex No. 2.7

SQL> select*from br;

BNAME	BCITY	ASSET
Cantonment	trichy	50000
rockfort	madurai	40000
t nagar	chennai	30000
kk nagar	salem	60000
rockfort	trichy	70000

//FIND AVERAGE ACCOUNT BALANCE AT A BRANCH:

SQL> select bname,avg(asset)from br where bname='cantonment'group by bname;

BNAME	AVG(ASSET)
Cantonment	50000

//FIND THE MINIMUM BALANCE AT A BRANCH:

SQL> select bname,min(asset)as minasset from br where bname='rockfort'group by bname;

BNAME	MINASSET
rockfort	40000

//FIND THE MAXIMUM BALANCE AT A BRANCH:

SQL> select bname,max(asset)as maxasset from br where bname='rockfort'group by bname;

BNAME	MAXASSET
rockfort	70000

//FIND THE TOTAL BALANCE AT A BRANCH:

SQL> select sum(asset)from br;

SUM(ASSET)
250000

//FIND THE NUMBER OF ACCOUNTS IN A BRANCH:

SQL> select count(*)bcity from br where bcity='trichy';

BCITY
2

SQL> commit;

Commit complete.

Ex No. 2.8 Aggregate Functions with group by and having clause

SQL> select*from br;

BNAME	BCITY	ASSET
cantonment	trichy	50000
rockfort	madurai	40000
t nagar	chennai	30000
kk nagar	salem	60000
rockfort	trichy	70000

//FIND THE AVERAGE ACCOUNT BALANCE AT EACH BRANCH:

SQL> select bname,avg(asset)from br group by bname;

BNAME	AVG(ASSET)
rockfort	55000
t nagar	30000
cantonment	50000
kk nagar	60000

//FIND BRANCH NAMES THOSE BRANCHES WHERE THE TOTAL BALANCE IS MORE THAN Rs. 30,000:

SQL> select bname,sum(asset)from br group by bname having sum(asset)>30000;

BNAME	SUM(ASSET)
rockfort	110000
cantonment	50000
kk nagar	60000

//FIND THE BRANCHES THOSE BRANCHES WHERE THE TOTAL ACCOUNT ARE MORE THAN 1:

```
SQL> select bname,count(asset)from br group by bname having count(asset)>=1;

BNAME          COUNT(ASSET)
-----------    --------------
rockfort              2
t nagar               1
cantonment            1
kk nagar              1

SQL> commit;
Commit complete.
```

Ex No. 2.9 Nested Sub-queries. Membership (in and not in)

SQL> select*from bo1;

CNAME	LNO
luffy	1002
sanji	1003
zoro	1004
nancy	1005

SQL> sele ct*from dep;

CNAME	ANO
thasu	12345
shafu	12346
ragu	12347
priya	12348
nami	12349
luffy	12350
sanji	12351
zoro	12352

8 rows selected.

//FIND ALL CUSTOMERS WHO HAVE BOTH A LOAN AND ACCOUNT AT A BANK:

SQL> select distinct cname from bo1 where cname in (select cname from dep);

CNAME

---

zoro

luffy

sanji

//FIND ALL CUSTOMERS WHO HAVE AN ACCOUNT BUT NO LOAN AT A BANK:

SQL> select distinct cname from bo1 where cname not in(select cname from dep);

1 rows selected

CNAME

---

nancy

SET COMPARISON (SOME,ALL)

SQL> select*from br;

BNAME	BCITY	ASSET
cantonment	trichy	50000
rockfort	madurai	40000
t nagar	chennai	30000
kk nagar	salem	60000
rockfort	trichy	70000

5 rows selected

SQL> select bname,asset from br where asset>some(select asset from br
where bcity='madurai');

BNAME	ASSET
cantonment	50000
kk nagar	60000
rockfort	70000

SQL> select bname,asset from br where asset>all(select asset from br where bcity='madurai');

BNAME	ASSET
cantonment	50000
kk nagar	60000
rockfort	70000

SUB-QUERY USED IN FROM CLAUSE

//FIND THE AVERAGE ASSETS OF THOSE BRANCHES WHERE THE AVERAGE ASSET IS GREATER THAN Rs. 40,000:

SQL> select bname,avg_asset from(select bname,avg(asset)as avg_asset from br group by bname) where avg_asset>40000;

BNAME AVG_ASSET

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

rockfort 55000

cantonment 50000

kk nagar 60000

//FIND THE MAXIMUM ACROSS ALL BRANCHES OF THE TOTAL BALANCE AT EACH BRANCH:

SQL> select max(sum_asset)from(select bname,sum(asset)as sum_asset from br group by bname);

MAX(SUM_ASSET)

110000

SQL> commit;

Commit complete.

Ex No.2.10 Views

SQL> select*from br;

BNAME	BCITY	ASSET
cantonment	trichy	50000
rockfort	madurai	40000
t nagar	chennai	30000
kk nagar	salem	60000
rockfort	trichy	70000

SQL> create view jk as(select bname,bcity from br);

View created.

SQL> select*from jk;

BNAME	BCITY
Cantonment	trichy
rockfort	madurai
t nagar	chennai
kk nagar	salem
rockfort	trichy

SQL> commit;

Commit complete.

Ex No. 2.11 Deletion

SQL> select*from act;

ANO	BNAME	AMOUNT
12345	cantonment	20000
12346	trichy	19000
12347	jk nagar	17000
12348	kk nagar	15000

SQL> select*from br;

BNAME	BCITY	ASSET
cantonment	trichy	50000
rockfort	madurai	40000
t nagar	chennai	30000
kk nagar	salem	60000
rockfort	trichy	70000

//DETELE THE TUPLES OF ALL ACCOUNTS WITH BALANCES BELOW THE AVERAGE AT THE BANK:

SQL> delete from act where bname='cantonment';

1 row deleted.

SQL> select*from act;

ANO	BNAME	AMOUNT
12346	trichy	19000
12347	jk nagar	17000
12348	kk nagar	15000

//DELETE ALL ACCOUNTS TUPLES AT EVERY BRANCH LOCATED IN A CITY:

SQL> delete from act where bname in(select bname from br where bcity='salem');

1 row deleted.

SQL> select*from act;

ANO	BNAME	AMOUNT
12346	trichy	19000
12347	jk nagar	17000

SQL> delete from act where amount<(select avg(amount)from act);

1 row deleted.

SQL> select*from act;

ANO	BNAME	AMOUNT
12346	trichy	19000

SQL> commit;

Commit complete.

Ex No. 2.12 Updates

SQL> select*from act;

ANO	BNAME	AMOUNT
12346	trichy	19000

//ALL BALANCES ARE TO THE INCREASED BY 1.1 PERCENT:

SQL> update act set amount=amount*1.1;

1 row updated.

SQL> select*from act;

ANO	BNAME	AMOUNT
12346	trichy	20900

//UPDATE WITH CASE STATEMENTS ALL ACCOUNTS WITH BALANCES OVER 10,000 RECEIVES 1.05 PERCENT INTEREST WHERE AS OTHERS RECEIVE 1.1 PERCENT:

SQL> update act set amount=case when amount<=10000 then amount*1.05 else amount*1.1 end;

1 row updated.

SQL> select*from act;

ANO	BNAME	AMOUNT
12346	trichy	22990

SQL> commit;

Commit complete.

Ex No. 2.13 Join Operations

```
SQL> create table s1(name varchar(5),regno number(5));

Table created.

SQL> desc s1;

Name                                      Null?    Type
----------------------------------------- -------- ----------------------------

NAME                                               VARCHAR2(5)

REGNO                                              NUMBER(5)

SQL> insert into s1 values('&name',&regno);

Enter value for name: thasu

Enter value for regno: 101

old   1: insert into s1 values('&name',&regno)

new   1: insert into s1 values('thasu',101)

1 row created.


SQL> select*from s1;

NAME  REGNO

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

thasu       101

ragu        102

shaju       103

priya       104

pree        105

shafu       109

jack        110

banu        111

8 rows selected.
```

SQL> create table markk(regno number(5),total number(5));
Table created.
SQL> insert into markk values(&regno,&total);
Enter value for regno: 101
Enter value for total: 456
old 1: insert into markk values(&regno,&total)
new 1: insert into markk values(101,456)
1 row created.

SQL> select*from markk;

REGNOTOTAL

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

101 456

102 425

103 303

104 466

105 355

106 402

107 325

108 225

8 rows selected.

//INNER JOIN:

SQL> select*from s1 join markk on s1.regno=markk.regno;

NAME REGNO REGNO TOTAL

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

thasu 101 101 456

ragu 102 102 425

shaju 103 103 303

priya 104 104 466

pree 105 105 355

5 rows selected

//LEFT OUTER JOIN:

SQL> select*from s1 left join markk on s1.regno=markk.regno;

NAME	REGNO	REGNO	TOTAL
thasu	101	101	456
ragu	102	102	425
shaju	103	103	303
priya	104	104	466
pree	105	105	355
banu	111	Null	Null
shafu	109	Null	Null
jack	110	Null	Null

8 rows selected.

//RIGHT OUTER JOIN:

SQL> select*from s1 right join markk on s1.regno=markk.regno;

NAME	REGNO	REGNO	TOTAL
thasu	101	101	456
ragu	102	102	425
shaju	103	103	303
priya	104	104	466
pree	105	105	355
Null	Null	106	402
Null	Null	108	225
Null	Null	107	325

8 rows selected.

PL/SQL PROCEDURE

Ex no. 3.1 Reverse the String

```
SQL> set serveroutput on
SQL> declare
  2  s1 varchar(20);
  3  s2 varchar(20);
  4  c number(20);
  5  begin
  6  s1:='&string';
  7  c:=length(s1);
  8  while c>0
  9  loop
 10  s2:=s2||substr(s1,c,1);
 11  c:=c-1;
 12  end loop;
 13  dbms_output.put_line('the given string is:'||s1);
 14  dbms_output.put_line('reverse string is:'||s2);
 15  end;
 16  /
```

OUTPUT:

```
Enter value for string: thasneem
old      6: s1:='&string';
new      6: s1:='thasneem';
the given string is:thasneem
reverse string is:meensaht

PL/SQL procedure successfully completed.
```

Ex No. 3.2 Student Mark Sheet Preparation

SQL> create table stud1(rollno int,name varchar(5),mark1 int,mark2 int,mark3 int,total int,avg int);

Table created.

SQL> desc stud1;

Name	Null?	Type
ROLLNO		NUMBER(38)
NAME		VARCHAR2(5)
MARK1		NUMBER(38)
MARK2		NUMBER(38)
MARK3		NUMBER(38)
TOTAL		NUMBER(38)
AVG		NUMBER(38)

SQL> insert into stud1 values(&rollno,'&name',&m1,&m2,&m3,",");
Enter value for rollno: 501
Enter value for name: ragu
Enter value for m1: 87
Enter value for m2: 76
Enter value for m3: 88
old 1: insert into stud1 values(&rollno,'&name',&m1,&m2,&m3,",")
new 1: insert into stud1 values(501,'ragu',87,76,88,",")
1 row created.

SQL> select*from stud1;

ROLLNO	NAME	MARK1	MARK2	MARK3	TOTAL	AVG
501	ragu	87	76	88		
502	thasu	90	75	96		
503	priya	88	87	77		
504	jack	69	89	76		
505	shafu	59	70	81		

```
SQL> set serveroutput on
SQL> declare
  2  cursor c1 is select*from stud1;
  3  x stud1 % rowtype;
  4  begin
  5  open c1;
  6  loop
  7  fetch c1 into x;
  8  exit when c1%notfound;
  9  x.total:=x.mark1+x.mark2+x.mark3;
 10  x.avg:=x.total/3;
 11  update stud1 set total=x.total,avg=x.avg where rollno=x.rollno;
 12  end loop;
 13  close c1;
 14  end;
 15  /
PL/SQL procedure successfully completed.
```

```
SQL> select*from stud1;
```

ROLLNO	NAME	MARK1	MARK2	MARK3	TOTAL	AVG
501	ragu	87	76	88	251	84
502	thasu	90	75	96	261	87
503	priya	88	87	77	252	84
504	jack	69	89	76	234	78
505	shafu	59	70	81	210	70

Ex No. 3.3 Pay Roll Preparation

SQL> create table emp11(eno number(3),ename varchar(10),bpay number(15),hra number(15),pf number(15),gpay number(15),netpay number(15));

Table created.

SQL> desc emp11;

Name	Null?	Type
ENO		NUMBER(3)
ENAME		VARCHAR2(10)
BPAY		NUMBER(15)
HRA		NUMBER(15)
PF		NUMBER(15)
GPAY		NUMBER(15)
NETPAY		NUMBER(15)

SQL> insert into emp11 values(&eno,'&ename',&bpay,",",",");

Enter value for eno: 101

Enter value for ename: luffy

Enter value for bpay: 13000

old 1: insert into emp11 values(&eno,'&ename',&bpay,",",",")

new 1: insert into emp11 values(101,'luffy',13000,",",",")

1 row created.

SQL> select*from emp11;

ENO	ENAME	BPAY	HRA	PF	GPAY	NETPAY
101	luffy	13000				
102	nami	23000				
103	zoro	34000				
104	sanji	16000				
105	robin	20000				

```
SQL> set serveroutput on
SQL> declare
  2  cursor a is select*from emp11;
  3  x emp11 % rowtype;
  4  begin
  5  open a;
  6  loop
  7  fetch a into x;
  8  exit when a%notfound;
  9  x.hra:=x.bpay*0.5;
 10  x.pf:=x.bpay*0.1;
 11  x.gpay:=x.bpay+x.hra;
 12  x.netpay:=x.gpay-x.pf;
 13  update emp11 set hra=x.hra,pf=x.pf,gpay=x.gpay,netpay=x.netpay where
eno=x.eno;
 14  end loop;
 15  close a;
 16  end;
 17  /
PL/SQL procedure successfully completed.
```



```
SQL> select*from emp11;
```

ENO	ENAME	BPAY	HRA	PF	GPAY	NETPAY
101	luffy	13000	6500	1300	19500	18200
102	nami	23000	11500	2300	34500	32200
103	zoro	34000	17000	3400	51000	47600
104	sanji	16000	8000	1600	24000	22400
105	robin	20000	10000	2000	30000	28000

Ex No.3.4 Find factorial number using recursive function

```
SQL> create or replace function fac(n number)return integer is
  2  begin
  3  if n=1 then return 1;
  4  else
  5  return n*fac(n-1);
  6  end if;
  7  end;
  8  /
```

Function created.

```
SQL> set serveroutput on
SQL> declare
  2  a number;
  3  b number:=&num;
  4  begin
  5  a:=fac(b);
  6  dbms_output.put_line('********************');
  7  dbms_output.put_line('Given number is:'||b);
  8  dbms_output.put_line('factorial  is:'||a);
  9  dbms_output.put_line('********************');
 10  end;
 11  /
```

OUTPUT:

```
Enter value for num: 3
old      3: b number:=&num;
new      3: b number:=3;
********************
Given number is:3
factorial is:6
********************
```

PL/SQL procedure successfully completed.

Ex No. 3.5 Program using Exceptional Handling

```
SQL> set serveroutput on
SQL> declare
  2  a number:=0;
  3  b number:=1;
  4  c number;
  5  begin
  6  dbms_output.put_line(a||"||b||");
  7  for i in 3..10 loop
  8  c:=a+b;
  9  dbms_output.put_line(c||");
 10  a:=b;
 11  b:=c;
 12  end loop;
 13  end;
 14  /
```

OUTPUT:

```
01
1
2
3
5
8
13
21
34
PL/SQL procedure successfully completed.
```