Program 11 **PART B**	**Consider the following schema:** **STUDENT MARKS PROGRAM**
Date:	STUDENT (USN, name, date_of_birth, branch, mark1, mark2, mark3, total, GPA) **Creating Tables, Inserting/Updating/Deleting Records in a Table, Saving (Commit).**

Note: To be written in record RULED page

Solution:

TABLE CREATION

```
SQL> CREATE TABLE STUDENT(
  2  USN VARCHAR(6),
  3  NAME VARCHAR(15),
  4  DATE_OF_BIRTH DATE,
  5  BRANCH CHAR(5),
  6  MARK1 NUMBER(3),
  7  MARK2 NUMBER(3),
  8  MARK3 NUMBER(3),
  9 TOTAL NUMBER(5),
 10  GPA DECIMAL(4,2),
 11  PRIMARY KEY (USN));
```

Table created.

B) Inserting Values

SQL> INSERT INTO STUDENT VALUES('BCA001','AISHU','28-DEC-1989','CS',70,60,65,195,7.5);

SQL> INSERT INTO STUDENT VALUES('BCA002','SNEHA','7-JAN-2000','ARTS',71,62,50,183,7.1);

SQL> INSERT INTO STUDENT VALUES('BCA003','RAM','25-NOV-2001','COM',80,70,50,200,7.7);

SQL> INSERT INTO STUDENT VALUES('BCA004','RAJ','5-FEB-2000','B.E',60,60,61,181,7.1);

SQL> INSERT INTO STUDENT VALUES('BCA005','KHAN','23-JUN-2002','B.A',70,75,55,200,7.7);

SQL> SELECT*FROM STUDENT;

a) Updating Record

SQL> UPDATE STUDENT SET NAME='SNEHA PANDIT' WHERE USN='BCA002';

SQL> SELECT*FROM STUDENT;

b) Deleting a Particular Record.

SQL> DELETE FROM STUDENT WHERE USN='BCA005';

SQL> SELECT*FROM STUDENT;

c) Committing

SQL> commit;

Commit complete.

Program 12 **PART B**	**1. Write a PLSQL program to perform Arithmetic operations.**
Date:	

SQL> SET SERVEROUTPUT ON;

```
SQL> declare
a int;
b  int;
c  int;
d  int;
e  int;
f  int;
begin
a := &a;
b := &b;
c := a+b;
d := a-b;
e := a*b;
f := a/b;
dbms_output.put_line('Addition of two numbers:' ||c);
dbms_output.put_line('Subtraction of two numbers:' ||d);
dbms_output.put_line('Multiplication of two numbers:' ||e);
dbms_output.put_line('Division of two numbers :' ||f);
end;
/
```

Program 13 **PART B**	**1. Consider the following schema:** STUDENT (USN, name, date_of_birth, branch, PER, GPA) **Execute the following queries:** **a. Find the GPA score of all the students.** **b. Find the students who born on a particular year of birth from the date_of_birth column.**
Date:	

Table creation

```
CREATE TABLE STUDENT(
  USN VARCHAR(6),
  NAME VARCHAR(15),
     DOB DATE,
     BRANCH VARCHAR(20),
   PERCENTAGE Decimal(5,2),
   GPA DECIMAL(4,2),
   PRIMARY KEY (USN));
```

Inserting recods

```
SQL > INSERT INTO STUDENT (usn, name, DOB, Branch, Percentage) VALUES
('BCA001','Akash','12-dec-2000','Computer Science',80.1);
SQL > INSERT INTO STUDENT (usn, name, DOB, Branch, Percentage) VALUES
('BCA002','Sneha','22-Jul-2001','Arts',70.9);
SQL > INSERT INTO STUDENT (usn, name, DOB, Branch, Percentage) VALUES
('BCA003','Prem','2-Jan-2000', 'ComputerScience',65.2);
SQL > INSERT INTO STUDENT (usn, name, DOB, Branch, Percentage) VALUES
('BCA004','Bikash','9-Sep-1999','Commerce',95.21);
SQL > INSERT INTO STUDENT (usn, name, DOB, Branch, Percentage) VALUES
('BCA005','Sandhya','17-Aug-2001','Arts',63.41);
SQL> select * from student;
```

a. Find the GPA score of all the students.

```
SQL> UPDATE STUDENT SET GPA=(PERCENTAGE/100)*4 ;
SQL> select * from student;
```

b. Find the students who born on a particular year of birth from the date_of_birth column.

```
SQL> SELECT * FROM STUDENT WHERE DOB between '01-Jan-2000' and '31-Dec-2000';
```

Program 14 **PART B** **Date:**	**1. Consider the following schema:** STUDENT (USN, name, date_of_birth, branch, PER, GPA) **Execute the following queries:** **a. List the students who are studying in a particular branch of study.** **b. Find the maximum GPA score of the student branch-wise.**

a. List the students who are studying in a particular branch of study.

```
SQL> SELECT * FROM STUDENT WHERE Branch='Arts';
```

b. Find the maximum GPA score of the student branch-wise.

```
SQL> SELECT Branch, Max(GPA) FROM STUDENT GROUP BY Branch;
```

Program 15 **PART B**	**1. Write a PLSQL program to find out to check whether a year is leap year or not.**
Date:	

SQL> SET SERVEROUTPUT ON;

A year is a leap year if following conditions are satisfied:
1) Year is multiple of 400
2) Year is multiple of 4 and not multiple of 100

```
--  true if the year is a multiple
  -- of 4 and not multiple of 100.

  -- OR year is multiple of 400.
```

```
DECLARE
Year NUMBER;
BEGIN
Year:=2022;
IF MOD(Year,4)=0 AND MOD(Year,100)!=0 OR MOD(Year,400)=0 THEN
DBMS_OUTPUT.PUT_LINE(Year ||' IS A LEAP YEAR');
ELSE
DBMS_OUTPUT.PUT_LINE(Year ||' IS NOT A LEAP YEAR');
END IF;
END;
/
```

Program 16 **PART B**	**1. A COWINAPPS database has a table with the following attributes.** COWINAPPS (Adhaarid:string, Name: string, Phno: integer, Age: Integer, Gender:char(6))
Date:	**a) Create a view named COWINVIEW that shows all the details of a person whose age is greater than 18.** **b) Display the view named COWINVIEW**

SQL> Create table COWINAPPS
(Aadharid Number,
Name varchar(10),
Gender char(6),
Age Number,
Phno Number,
primary key(Aadharid))

SQL> Insert into COWINAPPS values(11111,'Amal','M',37,9845917113);
SQL> Insert into COWINAPPS values(22222,'Sowmya','F',30,9400483767);
SQL> Insert into COWINAPPS values(33333,'Krithik','M',6,9985677413)
SQL> Insert into COWINAPPS values(44444,'vyshu','F',11,9800000000);
SQL> Insert into COWINAPPS values(55555,'Anu','F',19,93878327923);

a) Create a view named COWINVIEW that shows all the details of a person whose age is greater than 18.

SQL> Create view COWINVIEW AS select Adharid,Name,Age From COWINAPPS where Age > 18;

b) Display the view named COWINVIEW

SQL> select * from COWINVIEW;

Program 17 **PART B**	**1. Write a PLSQL program to find largest of two numbers.**
Date:	

```
SQL> SET SERVEROUTPUT ON;

SQL> DECLARE
 a int;
b  int;
begin
a := &a;
b := &b;
if(a>b) then
dbms_output.put_line('Greater numbers is' ||&a);
else
dbms_output.put_line('Greater numbers is' ||&b);
end if;
end;

/
```

Program 18 **PART B**	**Create the following tables with properly specifying Primary keys, Foreign keys and solve the following queries.** • BRANCH (Branchid-NUMBER, Branchname-VARCHAR, HOD-VARCHAR) • STUDENT (USN-NUMBER, Name, Address, Branchid-NUMBER, sem-NUMBER) • AUTHOR (Authorid-NUMBER, Authorname, Country, age-NUMBER) • BOOK (Bookid-NUMBER, Bookname, Authorid-NUMBER, Publisher, Branchid-NUMBER) • BORROW (USN-NUMBER, Bookid-NUMBER, Borrowed_Date) **a. Display the Book names in descending order of their names.** **b. Display the number of books written by each Author.**
Date:	

SOLUTION:

TABLE1:

```
Create table branch(
BRANCHID NUMBER,
BRANCHNAME VARCHAR(10),
HOD VARCHAR(15),
PRIMARY KEY(BRANCHID));

INSERT INTO BRANCH VALUES(100, 'BCA','DR.ALLI');
INSERT INTO BRANCH VALUES(101, 'BCOM','DR.AYESHA');
INSERT INTO BRANCH VALUES(102, 'BA','DR.SHILPA');
```

TABLE2:

```
Create table STUDENT(
USN NUMBER,
NAME VARCHAR(10),
ADDRESS VARCHAR(15),
BRANCHID NUMBER,
SEM NUMBER,
PRIMARY KEY(USN),
FOREIGN KEY(BRANCHID) REFERENCES BRANCH(BRANCHID));
```

```
INSERT INTO STUDENT VALUES(1, 'KIRAN','HEBBAL', 100,1);
INSERT INTO STUDENT VALUES(2, 'AMISHA','HEBBAL', 100,1);
INSERT INTO STUDENT VALUES(3, 'KEERTHI','SHIVAJINAGAR', 101,1);
INSERT INTO STUDENT VALUES(4, 'KARTHIK','SHIVAJINAGAR', 101,1);
INSERT INTO STUDENT VALUES(5, 'SATYA','SHIVAJINAGAR', 102,1);
INSERT INTO STUDENT VALUES(6, 'SNEHA','SHIVAJINAGAR', 102,1);
```

TABLE3:

```
Create table AUTHOR(
AUTHORID NUMBER,
AUTHORNAME VARCHAR(10),
COUNTRY VARCHAR(15),
AGE NUMBER,
PRIMARY KEY(AUTHORID));

INSERT INTO AUTHOR VALUES(11, 'SNEHA','INDIA', 36);
INSERT INTO AUTHOR VALUES(22, 'RICHA','INDIA', 39);
INSERT INTO AUTHOR VALUES(33, 'RAVI','MALASIYA', 45);
INSERT INTO AUTHOR VALUES(44, 'ILIYA','IRAN', 26);
```

TABLE4:

```
Create table BOOK(
BOOKID NUMBER,
BOOKNAME VARCHAR(10),
AUTHORID NUMBER,
PUBLISHER VARCHAR(15),
BRANCHID NUMBER,
PRIMARY KEY(BOOKID),
FOREIGN KEY(AUTHORID) REFERENCES AUTHOR(AUTHORID),
FOREIGN KEY(BRANCHID) REFERENCES BRANCH(BRANCHID));

INSERT INTO BOOK VALUES(1000, 'DBMS',11, 'HIMALAYA',100);
INSERT INTO BOOK VALUES(1001, 'C',22, 'SUBHAS',100);
INSERT INTO BOOK VALUES(1002, 'TALLY',33, 'HIMALAYA',101);
INSERT INTO BOOK VALUES(1003, 'FINANCE',33, 'KIRAN',101);
INSERT INTO BOOK VALUES(1004, 'JOURNALISM',44, 'SV PUBLISHERS',102);
INSERT INTO BOOK VALUES(1005, 'MEDIA',11, 'HIMALAYA',102);
```

TABLE5:

```
Create table BORROW(
USN NUMBER,
BOOKID NUMBER,
```

```
BORROWED_DATE DATE,
FOREIGN KEY(USN) REFERENCES STUDENT(USN),
FOREIGN KEY(BOOKID) REFERENCES BOOK(BOOKID)
);

INSERT INTO BORROW VALUES(1,1000,'18-JAN-22');
INSERT INTO BORROW VALUES(2,1001,'19-JAN-22');
INSERT INTO BORROW VALUES(2,1000,'19-JAN-22');
INSERT INTO BORROW VALUES(3,1002,'16-JAN-22');
INSERT INTO BORROW VALUES(2,1003,'3-JAN-22');
INSERT INTO BORROW VALUES(4,1003,'11-JAN-22');
```

a. Display the Book names in descending order of their names.

SQL> select bookname from book order by bookname desc;

b. Display the number of books written by each Author.

SQL> SELECT AUTHORID,COUNT(BOOKID) FROM BOOK GROUP BY AUTHORID;

Program 19 **PART B**	**Create the following tables with properly specifying Primary keys, Foreign keys and solve the following queries.** • BRANCH (Branchid-NUMBER, Branchname-VARCHAR, HOD-VARCHAR) • STUDENT (USN-NUMBER, Name, Address, Branchid-NUMBER, sem-NUMBER) • AUTHOR (Authorid-NUMBER, Authorname, Country, age-NUMBER) • BOOK (Bookid-NUMBER, Bookname, Authorid-NUMBER, Publisher, Branchid-NUMBER) • BORROW (USN-NUMBER, Bookid-NUMBER, Borrowed_Date) **a. List the details of Students who are all studying in 1 sem BCA.** **b. List the students who are not borrowed any books.**
Date:	

a. List the details of Students who are all studying in 1 sem BCA.

```
SQL> select BRANCH.BRANCHNAME,name,sem  from  student INNER JOIN branch ON
branch.branchid= student.branchid where branchname='BCA' and SEM='1';
```

b. List the students who are not borrowed any books.

```
SQL>select USN,NAME from STUDENT where USN NOT in(select BORROW.USN
from BORROW);
```

Program 20 **PART B**	**Create the following tables with properly specifying Primary keys, Foreign keys and solve the following queries.** • BRANCH (Branchid-NUMBER, Branchname-VARCHAR, HOD-VARCHAR) • STUDENT (USN-NUMBER, Name, Address, Branchid-NUMBER, sem-NUMBER) • AUTHOR (Authorid-NUMBER, Authorname, Country, age-NUMBER) • BOOK (Bookid-NUMBER, Bookname, Authorid-NUMBER, Publisher, Branchid-NUMBER) • BORROW (USN-NUMBER, Bookid-NUMBER, Borrowed_Date) **a. Display the student details who borrowed more than two books.** **b. Display the USN, Student name, Branch_name, Book_name, Author_name, Books_Borrowed_ Date of 1 sem BCA Students who borrowed books.**
Date:	

a. Display the student details who borrowed more than two books.

```
SQL> SELECT USN,COUNT(BOOKID) FROM BORROW GROUP BY USN
HAVING COUNT(*)>2;
```

b. Display the USN, Student name, Branch_name, Book_name, Author_name, Books_Borrowed_ Date of 1 sem BCA Students who borrowed books.

STUDENT→ S
BOOK→B
BRANCH→ BR
BORROW→BW
AUTHOR→ A

```
SQL>select distinct S.USN, S.NAME, BR.BRANCHNAME, B.BOOKNAME, A.AUTHORNAME
,BW.BORROWED_DATE from STUDENT S join branch Br on S.BRANCHID=Br.BRANCHID
join BORROW BW on S.USN=BW.USN join book B on B.BOOKID=BW.BOOKID join author A on
A.AUTHORID=B.AUTHORID where S.SEM=1 and br.BRANCHNAME='BCA';
```