**BCA**

**I SEMESTER**

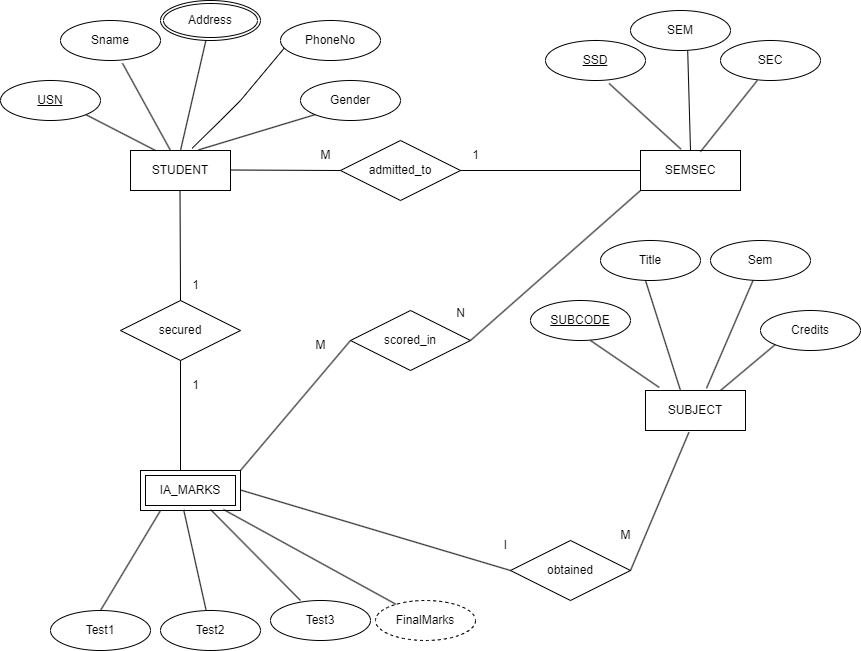
**DATABASE MANAGEMENT SYSTEM LAB**

Total Hours: 20 per batch Hours/Week: 2

Max Marks: 50 Credits: 2

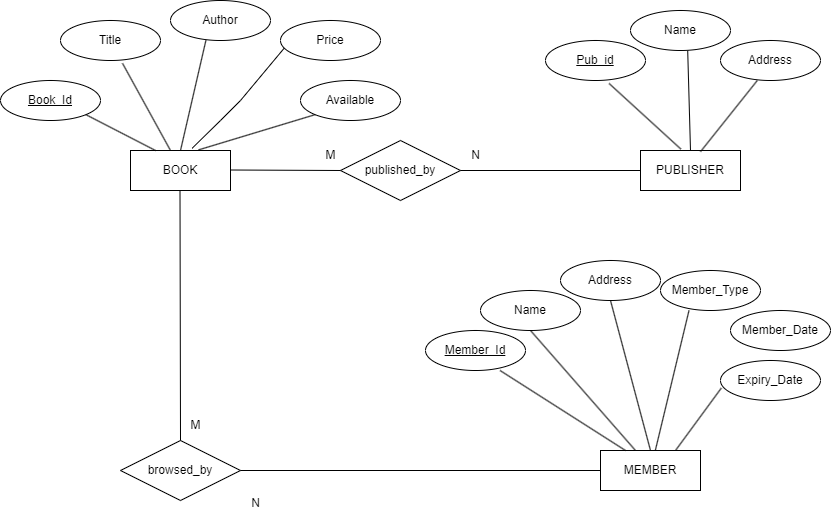
|  |  |
| --- | --- |
| Program 1 | Draw E-R diagram for a given scenario.  **COLLEGE DATABASE:**  STUDENT (USN, SName, Address, Phone, Gender)  SEM\_SEC (SSID, Sem, Sec)  CLASS (USN, SSID)  SUBJECT (Subcode, Title, Sem, Credits)  IA\_MARKS (USN, Subcode, SSID, Test1, Test2, Test3, FinalIA) |

**Solution:**



|  |  |
| --- | --- |
| Program 2 | Draw E-R diagram for a given scenario.  **LIBRARY DATABASE:**  BOOKS (Book\_Id, Author, Title, Price, Available)  PUBLISHER**(**Pub\_Id, Address ,Name)  MEMBER(Member\_Id,Name, Address, Member\_Type, Member\_date, Expiry\_Date) |

**Solution:**



|  |  |
| --- | --- |
| Program 3 | **The EMPLOYEE detail database has a table with the following attributes.** **EMPLOYEE** (Empno: int, Name: string, DOB: date, Phone\_no: int)   1. Create the above table. 2. Display table Structure. 3. Enter five tuples into the table. 4. Display all the tuples in EMPLOYEE table. |

**Solution:**

**a) Create the EMPLOYEE** **table.**

SQL> CREATE TABLE EMPLOYEE(

Empno int,

Name varchar(15),

DOB date,

Phone\_no number(12));

**b) Display table Structure.**

SQL> DESC EMPLOYEE;

Name Null? Type

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

EMPNO NUMBER(38)

NAME VARCHAR2(15)

DOB DATE

PHONE\_NO NUMBER(12)

**c) Display table Structure.**

SQL> insert into EMPLOYEE

values(1001,'Evelyn','02-aug-1993',8764563728);

1 row created.

SQL> insert into EMPLOYEE

values(1002,'Heather','24-sep-1996',9874563728);

1 row created.

SQL> insert into EMPLOYEE

values(1003,'Gregory','12-oct-1983',9994563728);

1 row created.

SQL> insert into EMPLOYEE

values(1004, 'Anthony','08-jun-1990',8974563728);

1 row created.

SQL> insert into EMPLOYEE

values(1005, 'Lindsey','17-jul-1979',7894563728);

1 row created.

**c) Display all the tuples in EMPLOYEE table.**

SQL> select \* from EMPLOYEE;

EMPNO NAME DOB PHONE\_NO

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

1001 Evelyn 02-AUG-93 8764563728

1002 Heather 24-SEP-96 9874563728

1003 Gregory 12-OCT-83 9994563728

1004 Anthony 08-JUN-90 8974563728

1005 Lindsey 17-JUL-79 7894563728

|  |  |
| --- | --- |
| Program 4 | Consider **EMPLOYEE** (Empno: int, Name: string, DOB: date, Phone\_no: int)  Perform the following:   1. Rename the table EMPLOYEE as EMP 2. Add a new column ‘Salary’ with not null constraint to the existing table EMP 3. Rename the column name ‘Name’ to ‘Emp\_name’ in EMP table 4. Delete the column ‘Salary’ from EMP table. 5. Truncate table to delete records. 6. Drop table |

**Creating EMPLOYEE table with primary key Empno:**

SQL> create table EMPLOYEE(

Empno int,

Name varchar(15),

DOB date,av

Phone\_no number(12),

primary key(Empno));

1. **Rename the table EMPLOYEE as EMP**

SQL> alter table EMPLOYEE rename to EMP;

Table altered.

Desc emp;

1. **Add a new column ‘Salary’ with not null constraint to the existing table EMP**

SQL> ALTER TABLE EMP ADD Salary DECIMAL(10, 2) NOT NULL;

Table altered.

Desc emp;

Insert into emp values( )

1. **Rename the column name ‘Name’ to ‘Emp\_name’ in EMP table**

SQL> ALTER TABLE EMP RENAME COLUMN Name TO Emp\_name;

Table altered.

SQL> DESC EMP;

Name Null? Type

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

EMPNO NUMBER(38)

EMP\_NAME VARCHAR2(15)

DOB DATE

PHONE

\_NO NUMBER(12)

SALARY NUMBER(10,2)

1. **Delete the column ‘Salary’ from EMP table.**

SQL> ALTER TABLE Emp1 DROP COLUMN Salary;

Table altered.

SQL> DESC Emp;

Name Null? Type

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

EMPNO NUMBER(38)

EMP\_NAME VARCHAR2(15)

DOB DATE

PHONE\_NO NUMBER(12)

1. **Truncate table to delete records.**

SQL> TRUNCATE TABLE Emp;

Table truncated.

SQL> SELECT \* FROM Emp;

no rows selected

1. **Drop table**

SQL> DROP TABLE Emp;

Table dropped.

**For Verification:**

SQL> DESC Emp;

ERROR:

ORA-04043: object EMP does not exist

|  |  |
| --- | --- |
| Program 5 | Perform the following: Viewing all Tables in a Database, Creating Tables (With and Without Constraints).  Consider the following tables:  **COMPANY**(Emp\_id: integer, Emp\_name: string, Gender: character)  **DEPT**(Dept\_id: integer, Dept\_name: string)  **SALARY**(Emp\_id: integer, Dept\_id: integer, Salary: integer)  Perform the following:   1. Creating Tables (specifying the primary keys and the foreign keys) 2. Enter 5 tuples in each table 3. Saving (Commit) and Undoing (rollback) 4. Viewing all Tables in a Database |

1. **Creating Tables (specifying the primary keys and the foreign keys)**

**COMPANY table:**

SQL> CREATE TABLE COMPANY (

Emp\_id INT PRIMARY KEY,

Emp\_name VARCHAR(255) NOT NULL,

Gender CHAR(1) NOT NULL

);

Table created

SQL> DESC COMPANY;

Name Null? Type

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

EMP\_ID NOT NULL NUMBER(38)

EMP\_NAME NOT NULL VARCHAR2(255)

GENDER NOT NULL CHAR(1)

**DEPT Table:**

SQL> CREATE TABLE DEPT (

Dept\_id INT PRIMARY KEY,

Dept\_name VARCHAR(255) NOT NULL

);

Table created.

SQL> DESC DEPT;

Name Null? Type

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

DEPT\_ID NOT NULL NUMBER(38)

DEPT\_NAME NOT NULL VARCHAR2(255)

**SALARY Table:**

CREATE TABLE SALARY (

Emp\_id INT NOT NULL,

Dept\_id INT NOT NULL,

Salary INT NOT NULL,

FOREIGN KEY (Emp\_id) REFERENCES COMPANY(Emp\_id),

FOREIGN KEY (Dept\_id) REFERENCES DEPT(Dept\_id)

);

SQL> DESC SALARY;

Name Null? Type

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

EMP\_ID NOT NULL NUMBER(38)

DEPT\_ID NOT NULL NUMBER(38)

SALARY NOT NULL NUMBER(38)

1. **Enter 5 tuples in each table**

**Inserting records into COMPANY table:**

SQL>INSERT INTO COMPANY

VALUES (1, 'Alice Johnson', 'F');

1 row created.

SQL>INSERT INTO COMPANY

VALUES (2, 'Bob Smith', 'M');

1 row created.

SQL>INSERT INTO COMPANY

VALUES (3, 'Catherine Zeta', 'F');

1 row created.

SQL>INSERT INTO COMPANY

VALUES (4, 'David Brown', 'M');

1 row created.

SQL>INSERT INTO COMPANY

VALUES (5, 'Eva Green', 'F');

1 row created.

**Inserting records into DEPT Table:**

SQL> INSERT INTO DEPT

VALUES (101, 'Human Resources');

1 row created.

SQL> INSERT INTO DEPT

VALUES (102, 'Finance');

1 row created.

SQL> INSERT INTO DEPT

VALUES (103, 'Engineering');

1 row created.

SQL> INSERT INTO DEPT

VALUES (104, 'Marketing');

1 row created.

SQL> INSERT INTO DEPT

VALUES (105, 'Sales');

1 row created.

**Inserting records into SALARY Table:**

SQL> INSERT INTO SALARY

VALUES (1, 101, 60000);

1 row created.

SQL> INSERT INTO SALARY

VALUES (2, 102, 55000);

1 row created.

SQL> INSERT INTO SALARY

VALUES (3, 103, 75000);

1 row created.

Saving (Commit) and Undoing (rollback)

**SQL> SAVEPOINT a;**

Savepoint created.

SQL> INSERT INTO SALARY VALUES (4, 104, 50000);

1 row created.

**SQL> SAVEPOINT b;**

Savepoint created.

SQL> INSERT INTO SALARY VALUES (5, 105, 65000);

1 row created.

SQL> SELECT \* FROM SALARY;

EMP\_ID DEPT\_ID SALARY

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

1 101 60000

2 102 55000

3 103 75000

4 104 50000

5 105 65000

**SQL> ROLLBACK TO a;**

Rollback complete.

SQL> SELECT \* FROM SALARY;

EMP\_ID DEPT\_ID SALARY

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

1 101 60000

2 102 55000

3 103 75000

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Program 6 | For a given data set perform the following Aggregate functions.   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | EmployeeName | Gender | Salary | DeptName | DeptHead | DeptLocation | | Sam | Male | 4500 | IT | John | London | | Pam | Female | 2300 | HR | Mike | Sydney | | Simon | Male | 1345 | IT | John | London | | Mary | Female | 2567 | HR | Mike | Sydney | | Todd | Male | 6890 | IT | John | London |  1. Create table EMPLOYEE 2. Insert above tuples. 3. Retrieve number of employees 4. Retrieve distinct Department name. |

**Solution:**

1. **Create table EMPLOYEE**

SQL> CREATE TABLE Employee(

EmployeeName VARCHAR(255) NOT NULL,

Gender CHAR(1) NOT NULL,

Salary DECIMAL(8,2) NOT NULL,

DeptName VARCHAR(255) NOT NULL,

DeptHead VARCHAR(255) NOT NULL,

DeptLocation VARCHAR(255) NOT NULL );

**b. Insert sample tuples**

SQL>INSERT INTO Employee VALUES ('Sam', 'Male', 4500, 'IT', 'John', 'London');

1 row created

SQL>INSERT INTO Employee VALUES ('Pam', 'Female', 2300, 'HR', 'Mike', 'Sydney');

1 row created

SQL>INSERT INTO Employee VALUES ('Simon', 'Male', 1345, 'IT', 'John', 'London');

1 row created

SQL>INSERT INTO Employee VALUES ('Mary', 'Female', 2567, 'HR', 'Mike', 'Sydney');

1 row created

SQL>INSERT INTO Employee VALUES ('Todd', 'Male', 6890, 'IT', 'John', 'London');

**c. Retrieve number of employees**

SQL>SELECT COUNT(\*) AS number\_of\_employees FROM Employee;

**d. Retrieve distinct department names**

SQL>SELECT DISTINCT Department FROM Employee;

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Program 7 | Consider the Employee database with following Schema   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | EmployeeName | Gender | Salary | DeptName | DeptHead | DeptLocation | | Sam | Male | 4500 | IT | John | London | | Pam | Female | 2300 | HR | Mike | Sydney | | Simon | Male | 1345 | IT | John | London | | Mary | Female | 2567 | HR | Mike | Sydney | | Todd | Male | 6890 | IT | John | London |   Find the sum of the salaries of all employees of the ‘IT’ department, as well as the maximum salary, the minimum salary, and the average salary in this department. |

**Solution:**

**(Note: Program 6 Table need to be created)**

SQL> SELECT SUM(salary) AS total\_salary,

MAX(salary) AS max\_salary,

MIN(salary) AS min\_salary,

AVG(salary) AS average\_salary FROM Employee WHERE department = 'IT';

|  |  |
| --- | --- |
| Program 8 | Consider the Company database with following Schema  **EMP** ( Eno : Number; Ename : String; Ebdate : Date; Address : Text; Gender : Text; Salary : Number; Deptno :Number)  **DEPT** (Dno :Number; Dname :String; Dlocation: String)  **PROJECT** ( Pno : Number; Pname : String ; Dnum : Number)  **WORKS\_ON** (Eno : Number; Pnum; Number; Hours : Number)  Execute the Following Queries   1. For each project, retrieve the project number, the project name, and the number of employee who work on that project (use GROUP BY) 2. Retrieve the name of employees who born in the year 1990's |

**Solution:**

**Create EMP table:**

SQL> CREATE TABLE EMP ( Eno NUMBER PRIMARY KEY,

Ename VARCHAR(255) NOT NULL,

Ebdate DATE,

Address varchar(50),

Gender varchar(3),

Salary NUMBER,

Deptno NUMBER );

Table created.

**Create DEPT table:**

SQL> CREATE TABLE DEPT ( Dno NUMBER PRIMARY KEY,

Dname VARCHAR(255) NOT NULL,

Dlocation VARCHAR(255) NOT NULL );

Table created.

**Create Project table:**

SQL> CREATE TABLE PROJECT (

Pno NUMBER PRIMARY KEY,

Pname VARCHAR(255) NOT NULL,

Dnum NUMBER,

FOREIGN KEY (Dnum) REFERENCES DEPT(Dno));

Table created

**Create works\_on table:**

SQL> CREATE TABLE WORKS\_ON (

Eno NUMBER,

Pno NUMBER,

Hours NUMBER,

PRIMARY KEY (Eno, Pno), -- Composite Primary Key

FOREIGN KEY (Eno) REFERENCES EMP(Eno),

FOREIGN KEY (Pno) REFERENCES PROJECT(Pno));

Table created

**a. For each project, retrieve the project number, the project name, and the number of employee who work on that project (use GROUP BY)**

SQL> SELECT P.Pno, P.Pname, COUNT(DISTINCT W.Eno) AS num\_employees FROM PROJECT P INNER JOIN WORKS\_ON W ON P.Pno = W.Pno GROUP BY P.Pno, P.Pname;

**b. Retrieve the name of employees who born in the year 1990's**

SQL>SELECT Ename

FROM EMP

WHERE

EXTRACT(YEAR FROM Ebdate) BETWEEN 1990 AND 1999;

(Note: EXTRACT function to extract the year from the Ebdate column.)

|  |  |
| --- | --- |
| Program 9 | Consider the Company database with following Schema  **EMP** ( Eno : Number; Ename : String; Ebdate : Date; Address : Text; Gender : Text; Salary : Number; Deptno :Number)  **DEPT** (Dno :Number; Dname :String; Dlocation: String)  **PROJECT** ( Pno : Number; Pname : String ; Dnum : Number)  **WORKS\_ON** (Eno : Number; Pnum; Number; Hours : Number)  Execute the Following Queries   1. For each Department that has more than five employees retrieve the department number and number of employees who are making salary more than 40000. 2. For each Department that has more than two employees retrieve the department number and number of employees who are making salary more than 40000. |

(NOTE: TO ENTER MINIMUM 6 TUPLES IN THE EMPLOYEE TABLE )

**CREATE TABLE EMP**

SQL>CREATE TABLE EMP

( Eno Number,

Ename VARCHAR(20),

Ebdate DATE,

Address varchar(100),

Gender VARCHAR(10),

Salary DECIMAL(10, 2),

Deptno NUMBER);

Table created.

**INSERT IN EMP TABLE**

SQL>INSERT INTO

EMP (Eno, Ename, Ebdate, Address, Gender, Salary, Deptno)

VALUES (1, 'John Doe','15-MAY-1990', '123 Main St, Anytown, USA', 'Male', 50000.00, 101);

1 row created.

SQL>INSERT INTO

EMP (Eno, Ename, Ebdate, Address, Gender, Salary, Deptno)

VALUES (2, 'Jane Smith', '20-AUG-2001', '456 Oak Ave, Somewhere, USA', 'Female', 60000.00, 102);

SQL>INSERT INTO

EMP (Eno, Ename, Ebdate, Address, Gender, Salary, Deptno)

VALUES (3, 'Michael Johnson', '10-MAR-1993', '789 Elm Rd, Nowhere, USA', 'Male', 55000.00, 101);

1 row created.

**DEPT TABLE CREATION**

SQL>CREATE TABLE DEPT (

Dno NUMBER,

Dname VARCHAR(20),

Dlocation VARCHAR(20));

Table created.

**INSERT VALUES IN DEPT TABLE**

SQL>INSERT INTO DEPT (Dno, Dname, Dlocation)

VALUES (101, 'HR Department', 'New York');

1 row created.

SQL>INSERT INTO DEPT (Dno, Dname, Dlocation)

VALUES (102, 'IT Department', 'San Francisco');

1 row created.

SQL>INSERT INTO DEPT (Dno, Dname, Dlocation)

VALUES (103, 'Sales Department', 'Chicago');

1 row created.

**CREATE TABLE PROJECT:**

SQL>CREATE TABLE PROJECT (

PNO NUMBER,

Pname VARCHAR(20),

Dnum NUMBER);

Table created.

**INSERT IN TABLE PROJECT:**

SQL>INSERT INTO PROJECT (Pno, Pname, Dnum)

VALUES (1, 'Project A', 101);

1 row created.

SQL>INSERT INTO PROJECT (Pno, Pname, Dnum)

VALUES (2, 'Project B', 102);

1 row created.

SQL>INSERT INTO PROJECT (Pno, Pname, Dnum)

VALUES (3, 'Project C', 103);

1 row created.

**CREATE A TABLE WORKS\_ON**

SQL>CREATE TABLE WORKS\_ON

(Eno NUMBER,

Pnum NUMBER,

Hours DECIMAL(5, 2));

Table created.

**INSERT IN WORKS\_ON TABLE**

SQL>INSERT INTO WORKS\_ON (Eno, Pnum, Hours)

VALUES (1, 1, 40.5);

1 row created.

SQL>INSERT INTO WORKS\_ON (Eno, Pnum, Hours)

VALUES (1, 2, 30);

1 row created.

SQL>INSERT INTO WORKS\_ON (Eno, Pnum, Hours)

VALUES (2, 1, 25.75);

1 row created.

SQL>INSERT INTO WORKS\_ON (Eno, Pnum, Hours)

VALUES (3, 3, 35);

1 row created.

**For each Department that has more than ONE employees retrieve the department number and number of employees who are making salary more than 40000.**

SQL>SELECT Deptno, COUNT(\*) AS NumEmployeesOver40k

FROM EMP

WHERE Salary > 40000

GROUP BY Deptno

HAVING COUNT(\*) > 5;

**For each Department that has more than two employees retrieve the department number and number of employees who are making salary more than 40000.**

SQL>SELECT Deptno, COUNT(\*) AS NumEmployeesOver40k

FROM EMP

WHERE Salary > 40000

GROUP BY Deptno

HAVING COUNT(\*) > 2;

|  |  |
| --- | --- |
| Program 10 | Consider the following schema:  **STUDENT** (USN, Name, DOB, Branch, Mark1, Mark2, Mark3, Total, GPA)  Execute the following queries   1. Creating Tables (With and Without Constraints), Inserting/Updating/Deleting Records in a Table, Saving (Commit) and Undoing (rollback) 2. List the students who are studying in a particular branch of study. 3. Find the maximum GPA score of the student branch-wise. |

1. **Create Table STUDENT**

SQL>CREATE TABLE STUDENT(

USN VARCHAR(15),

Name VARCHAR(15),

DOB DATE,

Branch CHAR(5),

Mark1 NUMBER(3),

Mark2 NUMBER(3),

Mark3 NUMBER(3),

Total NUMBER(5),

GPA DECIMAL(4,2),

PRIMARY KEY (USN));

Table created.

1. **Insert into STUDENT Table**

SQL>INSERT INTO STUDENT VALUES

('BCA001','AISHU','28-DEC-1989','CS',70,60,65,195,7.5);

1 row Created.

SQL>INSERT INTO STUDENT VALUES

('BCA002','SNEHA','7-JAN-2000','ARTS',71,62,50,183,7.1);

1 row Created.

SQL>INSERT INTO STUDENT VALUES

('BCA003','RAM','25-NOV-2001','COM',80,70,50,200,7.7);

1 row Created.

SQL>INSERT INTO STUDENT VALUES

('BCA004','RAJ','5-FEB-2000','B.E',60,60,61,181,7.1);

1 row Created.

SQL>INSERT INTO STUDENT VALUES

('BCA005','KHAN','23-JUN-2002','B.A',70,75,55,200,7.7);

1 row created.

SQL>INSERT INTO STUDENT VALUES

('BCA008','RAJAT','23-JUN-2002','CS',70,75,55,200,6.6);

1 row created.

1. **Display the Table Content**

SQL>SELECT \* FROM STUDENT;

(NOTE: use command SET PAGESIZE 20;

SET LINESIZE 120;)

1. **Update the Table Content**

SQL>UPDATE STUDENT

SET NAME='SNEHA PANDIT'

WHERE USN='BCA002';

1 row updated.

SQL>SELECT \* FROM STUDENT;

1. **DELETE the table Content**

SQL>DELETE FROM STUDENT

WHERE USN='BCA005';

1 row deleted.

SQL>SELECT \* FROM STUDENT;

1. **Commit(SAVE THE DATA)**

COMMIT complete.

SQL>DELETE FROM STUDENT WHERE USN='BCA004';

1 row deleted.

SQL>SELECT \* FROM STUDENT;

1. **ROLLBACK(UNDO THE CHANGES)**

SQL>ROLLBACK;

Rollback complete.

SQL> SELECT \* FROM STUDENT;

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

SQL>SELECT \* FROM STUDENT

WHERE Branch='CS';

1. **Find the maximum GPA score of the student branch-wise.**

SQL>SELECT Branch, MAX(GPA) AS MaxGPA

FROM STUDENT

GROUP BY Branch;

|  |  |
| --- | --- |
| Program 11 | Consider the following schema:  **STUDENT (USN, Name, DOB, Branch, Mark1, Mark2, Mark3, Total, GPA)**  Execute the following queries:   * 1. Find the GPA score of all the students.   2. Find the students who born on a particular year of birth from the ‘DOB’ column. |

**(NOTE: From Program 10, Queries a,b,c to be executed for the further operations)**

1. **Find the GPA score of all the students**

SQL>SELECT GPA

FROM STUDENT;

1. **Find the students who born on a particular year of birth from the ‘DOB’ column**

SQL>SELECT \* FROM STUDENT

WHERE DOB between '01-Jan-2000' and '31-Dec-2000';

|  |  |
| --- | --- |
| Program 12 | Create the following tables with properly specifying Primary keys, foreign keys andsolve the following queries.  **BRANCH** (Branch\_id, Branchname, HOD)  **STUDENT** (USN, Name, Address, Branch\_id, Sem)  **BOOK** (Book\_id, Bookname, Author\_id, Publisher, Branch\_id)  **AUTHOR** (Author\_id, Authorname, Country, Age)  **BORROW** (USN, Book\_id, Borrowed\_Date)  Execute the following queries   1. Display the Book names in descending order of their names. 2. Display the number of books written by each Author. 3. List the details of Students who are all studying in 2nd sem BCA. 4. List the students who are not borrowed any books. 5. Display the student details who borrowed more than two books. |

**To Create Tables:**

**BRANCH table**

SQL>CREATE TABLE BRANCH(

Branch\_id Number PRIMARY KEY,

Branchname VARCHAR(20),

HOD VARCHAR(20) );

**STUDENT table**

SQL>CREATE TABLE STUDENT1 (

USN VARCHAR(10) PRIMARY KEY,

Name VARCHAR(20),

Address VARCHAR(20),

Branch\_id Number,

Sem Number,

FOREIGN KEY (Branch\_id) REFERENCES BRANCH(Branch\_id));

**AUTHOR table**

SQL>CREATE TABLE AUTHOR (

Author\_id Number PRIMARY KEY,

Authorname VARCHAR(20),

Country VARCHAR(20),

Age Number

);

**BORROW table**

SQL>CREATE TABLE BORROW (

USN VARCHAR(20),

Book\_id Number,

Borrowed\_Date DATE,

PRIMARY KEY (USN, Book\_id),

FOREIGN KEY (USN) REFERENCES STUDENT(USN),

FOREIGN KEY (Book\_id) REFERENCES BOOK(Book\_id) );

**Insert Values in the Branch Table:**

SQL>INSERT INTO BRANCH (Branch\_id, Branchname, HOD)

VALUES (1, 'Computer Science', 'John Doe');

1 row created.

SQL>INSERT INTO BRANCH (Branch\_id, Branchname, HOD)

VALUES (2, 'Electrical Engineering', 'Jane Smith');

1 row created.

SQL>INSERT INTO BRANCH (Branch\_id, Branchname, HOD)

VALUES (3, 'Mechanical Engineering', 'Mike Johnson');

1 row created.

**Insert Values in the STUDENT Table**

SQL>INSERT INTO STUDENT1 (USN, Name, Address, Branch\_id, Sem)

VALUES ('1MS17CS001', 'Alice', '123 Main St', 1, 3);

1 row created.

INSERT INTO STUDENT1 (USN, Name, Address, Branch\_id, Sem)

VALUES ('1MS17ME003', 'Charlie', '789 Oak St', 3, 4);

1 row created.

**Insert Values in the AUTHOR Table**

SQL>INSERT INTO AUTHOR (Author\_id, Authorname, Country, Age)

VALUES (1, 'J.K. Rowling', 'UK', 56);

1 row created.

SQL>INSERT INTO AUTHOR (Author\_id, Authorname, Country, Age)

VALUES (2, 'Stephen King', 'USA', 75);

1 row created.

SQL>INSERT INTO AUTHOR (Author\_id, Authorname, Country, Age)

VALUES (3, 'Haruki Murakami', 'Japan', 73);

1 row created.

**Insert Values in the BORROW Table**

SQL>INSERT INTO BORROW (USN, Book\_id, Borrowed\_Date)

2 VALUES ('1MS17CS001', 101, '28-dec-2000');

1 row created.

SQL>INSERT INTO BORROW (USN, Book\_id, Borrowed\_Date)

2 VALUES ('1MS17ME003', 103, '20-jun-2001');

1 row created.

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

SQL>SELECT Bookname

FROM BOOK

ORDER BY Bookname DESC;

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

SQL>SELECT Author\_id, COUNT(\*) AS Num\_of\_Books

FROM BOOK

GROUP BY Author\_id;

List the details of Students who are all studying in 3rdsem Computer Science

SQL>SELECT \*

FROM STUDENT1

WHERE Sem = 3 AND Branch\_id = (SELECT Branch\_id FROM BRANCH WHERE Branchname = 'Computer Science');

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

SQL>SELECT \* FROM STUDENT1

WHERE USN

NOT IN (SELECT DISTINCT USN FROM BORROW);

**Display the student details who borrowed more than two books**

SQL>SELECT s.\*

FROM STUDENT1 s

WHERE s.USN IN (

SELECT b.USN

FROM (

SELECT USN, COUNT(\*) AS Num\_of\_Borrowings

FROM BORROW

GROUP BY USN

) b

WHERE b.Num\_of\_Borrowings > 2

);

|  |  |
| --- | --- |
| Program 13 | Write a PLSQL program to perform Arithmetic operations |

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 14 | Write a PLSQL program to find out to check whether a year is leap year or not. |

SQL> SET SERVEROUTPUT ON;

SQL> 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 15 | Write a PLSQL program to find largest of two numbers. |

SQL> SET SERVEROUTPUT ON;

SQL>DECLARE

num1 number;

num2 number;

greatest number;

BEGIN

num1 := &num1;

num2 := &num2;

IF num1 > num2 THEN

greatest := num1;

ELSE

greatest := num2;

END IF;

DBMS\_OUTPUT.PUT\_LINE('The greatest number is: ' || greatest);

END;

/