**DATABASE APPLICATIONS LABORATORY**

**Subject Code: 10CSL57 I.A. Marks : 25**

**Hours/Week : 03 Exam Hours: 03**

**Total Hours : 42 Exam Marks: 50**

1. Consider the following relations:

Student (*snum:* integer, *sname:* string, *major:* string, *level:* string, *age:* integer)

Class (*name:* string, *meets at:* string, *room:* string, *d:* integer)

Enrolled (*snum:* integer, *cname:* string)

Faculty (fi*d*: integer, *fname:* string, *deptid:* integer)

The meaning of these relations is straightforward; for example,Enrolled has one record per student-class pair such that the studentis enrolled in the class. Level is a two character code with 4 different values (example: Junior: JR etc)

Write the following queries in SQL. No duplicates should be printedin any of the answers.

i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by Prof. Harshith

ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

iii. Find the names of all students who are enrolled in two classes that meet at the same time.

iv. Find the names of faculty members who teach in every room in which some class is taught.

v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

2. The following relations keep track of airline flight information:

Flights (*no:* integer, *from:* string, *to:* string, *distance:* integer,

*Departs:* time, *arrives:* time, *price:* real)

Aircraft (*aid:* integer, *aname:* string, *cruisingrange:* integer)

Certified (*eid:* integer, *aid:* integer)

Employees (*eid:* integer, *ename:* string, *salary:* integer)

Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified for some aircraft,and only pilots are certified to fly.

Write each of the following queries in SQL.

i. Find the names of aircraft such that all pilots certified to

operate them have salaries more than Rs.80, 000.

ii. For each pilot who is certified for more than three aircrafts,

find the *eid* and the maximum *cruisingrange* of the aircraft

for which she or he is certified.

iii. Find the names of pilots whose *salary* is less than the price

of the cheapest route from Bengaluru to Frankfurt.

iv. For all aircraft with *cruisingrange* over 1000 Kms, .find the

name of the aircraft and the average salary of all pilots

certified for this aircraft.

v. Find the names of pilots certified for some Boeing aircraft.

vi. Find the *aid*s of all aircraft that can be used on routes from

Bengaluru to New Delhi.

3. Consider the following database of student enrollment in courses &

books adopted for each course.

STUDENT (regno: string, name: string, major: string, bdate:date)

COURSE (course #:int, cname:string, dept:string)

ENROLL ( regno:string, course#:int, sem:int, marks:int)

BOOK \_ ADOPTION (course# :int, sem:int, book-ISBN:int)

TEXT (book-ISBN:int, book-title:string, publisher:string,author:string)

i. Create the above tables by properly specifying the primarykeys and the foreign keys.

ii. Enter at least five tuples for each relation.

iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department.

iv. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the ‘CS’ department that use more than two books.

v. List any department that has *all* its adopted books published by a specific publisher.

vi. Generate suitable reports.

vii. Create suitable front end for querying and displaying the results.

4. The following tables are maintained by a book dealer.

AUTHOR (author-id:int, name:string, city:string, country:string)

PUBLISHER (publisher-id:int, name:string, city:string,country:string)

CATALOG (book-id:int, title:string, author-id:int, publisher-id:int, category-id:int, year:int, price:int)

CATEGORY (category-id:int, description:string)

ORDER-DETAILS (order-no:int, book-id:int, quantity:int)

i. Create the above tables by properly specifying the primary keys and the foreign keys.

ii. Enter at least five tuples for each relation.

iii. Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than the average price of the books in the catalog and the year of

publication is after 2000.

iv. Find the author of the book which has maximum sales.

v. Demonstrate how you increase the price of books published

by a specific publisher by 10%.

vi. Generate suitable reports.

vii. Create suitable front end for querying and displaying the

results.

5. Consider the following database for a banking enterprise

BRANCH(branch-name:string, branch-city:string, assets:real)

ACCOUNT(accno:int, branch-name:string, balance:real)

DEPOSITOR(customer-name:string, accno:int)

CUSTOMER(customer-name:string, customer-street:string,

customer-city:string)

LOAN(loan-number:int, branch-name:string, amount:real)

BORROWER(customer-name:string, loan-number:int)

i. Create the above tables by properly specifying the primary keys and the foreign keys

ii. Enter at least five tuples for each relation

iii. Find all the customers who have at least two accounts at the *Main* branch.

iv. Find all the customers who have an account at *all* the branches located in a specific city.

v. Demonstrate how you delete all account tuples at every branch located in a specific city.

vi. Generate suitable reports.

vii. Create suitable front end for querying and displaying the results.

***Instructions:***

1. The exercises are to be solved in an RDBMS environment like Oracle or DB2.

2. Suitable tuples have to be entered so that queries are executed correctly.

3. Front end may be created using either VB or VAJ or any other similar tool.

4. The student need not create the front end in the examination. The results of the queries may be displayed directly.

5. Relevant queries other than the ones listed along with the exercises may also be asked in the examination.

6. Questions must be asked based on lots.

**EXPERIMENT 1:**

1. Consider the following relations:

Student (*snum:* integer, *sname:* string, *major:* string, *level:* string, *age:* integer)

Class (*name:* string, *meets at:* string, *room:* string, *d:* integer)

Enrolled (*snum:* integer, *cname:* string)

Faculty (fi*d*: integer, *fname:* string, *deptid:* integer)

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class. Level is a two character code with 4 different values (example: Junior: JR etc)

Write the following queries in SQL. No duplicates should be printed in any of the answers.

i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by Prof. Harshith

ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

iii. Find the names of all students who are enrolled in two classes that meet at the same time.

iv. Find the names of faculty members who teach in every room in which some class is taught.

v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

**RELATIONAL SCHEMA DIAGRAM**

**STUDENT:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Snum** | **Sname** | **Major** | **Level** | **age** |

**CLASS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Meets-at** | **Room** | **d** |

**ENROLLED:**

|  |  |
| --- | --- |
| **Snum** | **Cname** |

**FACLTY:**

|  |  |  |
| --- | --- | --- |
| **Fid** | **Fname** | **deptid** |

**CREATING TABLES:**

SQL> CREATE TABLE STUDENT(snum Number Primary key, sname varchar(20),

major varchar(10),level1 varchar(10), age Number);

SQL> CREATE TABLE CLASS( Name varchar(20) primary key,meetsat varchar(10),

Room varchar(10),d Number,foreign key fk1 reference

Faculty(fid));

SQL> CREATE TABLE ENROLLED (snum Number,cname varchar(20),

constraint pk1 primary key(snum,cname),

constraint fk2 foreign key(snum) references Student(snum),

constraint fk3 foreign key(Cname) references Enrolled(name));

SQL> CREATE TABL FACULTY(fid Number primary key,Fname varchar(30),

deptid Number);

**INSERTING VALUES:**

SQL> INSERT INTO STUDENT VALUES (&snum,’&sname’, ‘&major’,’&level1’, &age);

SQL> INSERT INTO STUDENT VALUES(‘&Name’,’&meetsat’,’&Room’,&d);

SQL> INSERT INTO STUDENT VALUES ENROLLED (&snum,’&cname’);

SQL> INSERT INTO STUDENT VALUES (&snum,’&cname’);

**DISPLAYING TABLES:**

SQL> Select \* from Student;

SNUM SNAME MAJOR AGE LEVEL1

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

1 Tom CS 23 JR

2 Sri IS 21 SR

3 Alex IS 25 SR

4 Tony IS 19 JR

5 John IS 20 JR

SQL> select \* from class;

NAME MEETSAT ROOM D

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

III Semester 9:00 R128 5

IV Sem 9:55 R128 2

V Sem 11:10 R128 3

VII Sem 12:00 R128 3

I Sem 9:00 408 5

SQL> select \* from Enrolled;

SNUM CNAME

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

5 III Semester

1 IV Sem

4 V Sem

3 VII Sem

2 V Sem

5 I Sem

6 rows selected.

SQL> select \* from Faculty;

FID FNAME DEPTID

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

1 Mr. Selvin Peter 1

2 I Harshith 1

3 Ms. madhuri 1

4 Mr. Kashinath 1

5 Mr. Doddegowda 2

* SQL>SELECT DISTINCT S.Sname

FROM Student S, Class C, Enrolled E, Faculty F

WHERE S.snum = E.snum AND E.cname = C.name

AND C.d = F.fid AND

F.fname = 'I Harshith' AND S.level1 = 'JR';

**OUTPUT:**

SNAME

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

Tom

* SQL>SELECT C.name

FROM Class C

WHERE C.room = 'R128'

OR C.name IN (SELECT E.cname

FROM Enrolled E

GROUP BY E.cname

HAVING COUNT (\*) >= 5);

**OUTPUT:**

NAME

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

III Semester

IV Sem

V Sem

VII Sem

* SQL>SELECT DISTINCT S.sname

FROM Student S

WHERE S.snum IN (SELECT E1.snum

FROM Enrolled E1, Enrolled E2, Class C1, Class C2

WHERE E1.snum = E2.snum AND E1.cname <> E2.cname

AND E1.cname = C1.name AND E2.cname = C2.name

AND C1.meetsat = C2.meetsat);

**OUTPUT:**

SNAME

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

John

* SELECT DISTINCT F.fname

FROM Faculty F

WHERE NOT EXISTS ((SELECT \* FROM Class C)

EXCEPT

(SELECT C1.room FROM Class C1 WHERE C1.d = F.fid ))

* SQL> SELECT DISTINCT F.fname

FROM Faculty F

WHERE 5 > (SELECT count(E.snum) FROM Class C, Enrolled E

WHERE C.name = E.cname AND C.d = F.fid);

**OUTPUT:**

FNAME

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

I Harshith

Mr. Doddegowda

Mr. Kashinath

Mr. Selvin Peter

Ms. madhuri

**EXPERIMENT 2:**

2. The following relations keep track of airline flight information:

Flights (*no:* integer, *from:* string, *to:* string, *distance:* integer,

*Departs:* time, *arrives:* time, *price:* real)

Aircraft (*aid:* integer, *aname:* string, *cruisingrange:* integer)

Certified (*eid:* integer, *aid:* integer)

Employees (*eid:* integer, *ename:* string, *salary:* integer)

Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified for some aircraft,and only pilots are certified to fly.

Write each of the following queries in SQL.

i. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80, 000.

ii. For each pilot who is certified for more than three aircrafts, find the *eid* and the maximum *cruisingrange* of the aircraft for which she or he is certified.

iii. Find the names of pilots whose *salary* is less than the price

iv. For all aircraft with *cruisingrange* over 1000 Kms, .find the

name of the aircraft and the average salary of all pilots

certified for this aircraft.

v. Find the names of pilots certified for some Boeing aircraft.

vi. Find the *aid*s of all aircraft that can be used on routes from

Bengaluru to New Delhi.

**RELATIONAL SCHEMA DIAGRAM**

**FLIGHTS:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **From** | **To** | **Distance** | **Departs** | **Arrives** | **price** |

**AIRCRAFT:**

|  |  |  |
| --- | --- | --- |
| **Aid** | **Aname** | **Cruisingrange** |

**CERTIFIED:**

|  |  |
| --- | --- |
| **Eid** | **Aid** |

**EMPLOYEES:**

|  |  |  |
| --- | --- | --- |
| **Eid** | **Ename** | **Salary** |

**TABLE CREATION:**

SQL> create table Aircraft(aid number,aname varchar(20),

crusingrange number);

Table created.

SQL> create table Employee(eid number primary key,ename varchar(30),

salary number);

Table created.

SQL> create table Employee(eid number primary key,

ename varchar(30),salary number);

Table created.

SQL> create table Certified(eid number,aid number,

constraint pk101 primary key(eid,aid));

Table created.

**INSERTING VALUES IN FLIGHT TABLE:**

SQL> insert into Flight values(&no,'&from1','&to1',&distance,

'&Departs','&arrives',&price);

Enter value for no: 100 5000 9:00am 10:00pm

Enter value for from1:L.A.

Enter value for to1:Honolulu

Enter value for distance:5000

Enter value for Departs:9:00am

Enter value for arrives:10:00pm

Enter value for price:100000000

1 row created.

**INSERTING VALUES IN AIRCRAFT TABLE:**

SQL> insert into Aircraft values(&aid,'&aname',&crusingrange);

Enter value for aid: 1

Enter value for aname: Boeing A51

Enter value for crusingrange: 50

old 1: insert into Aircraft values(&aid,'&aname',&crusingrange)

new 1: insert into Aircraft values(1,'Boeing A51',50)

1 row created.

...

**INSERTING VALUES IN EMPLOYEE TABLE:**

SQL> insert into Employee values(&eid,'&ename',&salary);

Enter value for eid: 10

Enter value for ename: Tom

Enter value for salary: 125000

old 1: insert into Employee values(&eid,'&ename',&salary)

new 1: insert into Employee values(10,'Tom',125000)

1 row created.

...

**INSERTING VALUES IN CERTIFIED TABLE:**

SQL> insert into certified values(&eid,&aid);

Enter value for eid: 10

Enter value for aid: 2

old 1: insert into certified values(&eid,&aid)

new 1: insert into certified values(10,2)

1 row created.

**DISPLAYING TABLES:**

SQL> select \* from flights;

NO FROM1 TO1 DISTANCE DEPARTS ARRIVES PRICE

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

100 L.A. Honolulu 5000 9:00am 10:00pm 100000000

101 New Delhi New York 10000 9:00am 12:00pm 12345600

102 L.A. Chicago 500 3:30pm 9:30am 12500000

103 L.A. Chicago 500 7:15am 1:00pm 17500900

SQL> select \* from aircraft;

AID ANAME CRUSINGRANGE

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

1 Boeing A51 50

2 Air Bus 520 2000

3 Boeing A72 1500

4 Air Bus 400 1200

5 Boeing 477 1900

SQL> select \* from certified;

EID AID

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

10 2

11 1

3 4

4 5

13 5

14 4

11 5

10 1

14 5

11 2

15 2

13 2

11 3

13 rows selected.

SQL> select \* from employee;

EID ENAME SALARY

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

10 Tom 125000

11 Sri 150000

12 Xavier 130000

13 Sri 75000

14 Sri 190000

15 Thomas 48000

6 rows selected.

**QUERY:**

* SQL>SELECT DISTINCT A.aname

FROM Aircraft A

WHERE A.Aid IN (SELECT C.aid FROM Certified C, Employee E

WHERE C.eid = E.eid AND

NOT EXISTS (SELECT \*

FROM Employee E1 WHERE E1.eid = E.eid AND E1.salary >80000));

**OUTPUT:**

ANAME

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

Boeing 477

* SQL>SELECT C.eid, MAX (A.crusingrange)

FROM Certified C, Aircraft A

WHERE C.aid = A.aid

GROUP BY C.eid

HAVING COUNT (\*) > 3;

**OUTPUT:**

EID MAX(A.CRUSINGRANGE)

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

11 2000

* SQL> SELECT DISTINCT E.ename

FROM Employee E

WHERE E.salary < ( SELECT MIN (F.price)

FROM Flights F WHERE F.from1 = 'LA' AND F.to1 = 'Honolulu' );

**OUTPUT:**

ENAME

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

Sri

Thomas

Tom

Xavier

* Observe that *aid* is the key for Aircraft, but the question asks for aircraft names;

we deal with this complication by using an intermediate relation Temp:

SQL> SELECT Temp.name, Temp.AvgSalary

FROM ( SELECT A.aid, A.aname AS name, AVG (E.salary) AS AvgSalary

FROM Aircraft A, Certified C, Employee E

WHERE A.aid = C.aid AND C.eid = E.eid AND A.crusingrange > 1000

GROUP BY A.aid, A.aname ) Temp;

**OUTPUT:**

NAME AVGSALARY

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

Air Bus 520 99500

Boeing A72 150000

Air Bus 400 190000

Boeing 477 138333.333

* SQL> SELECT DISTINCT E.ename

FROM Employee E, Certified C, Aircraft A

WHERE E.eid = C.eid AND C.aid = A.aid AND

A.aname = 'Boeing 477';

**OUTPUT:**

ENAME

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

Sri

* SQL> SELECT A.aid FROM Aircraft A

WHERE A.crusingrange > ( SELECT MIN (F.distance)

FROM Flights F

WHERE F.from1 = 'L.A.' AND F.to1 = 'Chicago' );

**OUTPUT:**

AID

----------

2

3

4

5

**EXPERIMENT 3:**

3. Consider the following database of student enrollment in courses &

books adopted for each course.

STUDENT (regno: string, name: string, major: string, bdate:date)

COURSE (course #:int, cname:string, dept:string)

ENROLL ( regno:string, course#:int, sem:int, marks:int)

BOOK \_ ADOPTION (course# :int, sem:int, book-ISBN:int)

TEXT (book-ISBN:int, book-title:string, publisher:string,author:string)

i. Create the above tables by properly specifying the primarykeys and the foreign keys.

ii. Enter at least five tuples for each relation.

iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department.

iv. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the ‘CS’ department that use more than two books.

v. List any department that has *all* its adopted books published by a specific publisher.

vi. Generate suitable reports.

vii. Create suitable front end for querying and displaying the results.

**RELATIONAL SCHEMA DIAGRAM**

**STUDENT:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Regno** | **Name** | **Major** | **Bdate** |

**COURSE:**

|  |  |  |
| --- | --- | --- |
| **Course#** | **Cname** | **dept** |

**ENROLL:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Regno** | **Course#** | **Sem** | **Marks** | **ISBN** |

**BOOK-ADOPTION:**

|  |  |  |
| --- | --- | --- |
| **Course#** | **Sem** | **Book-ISBN** |

**TEXT:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Book-ISBN** | **Book-title** | **Publisher** | **author** |

* Create the above tables by properly specifying the primary keys and the foreign keys.

SQL> create table student

(

regno varchar(15),

name varchar(15),

major varchar(15),

bdate date,

constraint pkey12 primary key(regno)

);

Table created.

SQL> create table course

(

courseno int,

cname varchar(15),

dept varchar(15),

constraint pkey13 primary key(courseno)

);

Table created.

SQL> create table text

(

isbn int,

booktitle varchar(15),

publisher varchar(15),

author varchar(15),

constraint pkey14 primary key(isbn)

);

Table created.

SQL> create table enroll

(

regno varchar(15),

courseno int,

sem int,

isbn int,

constraint pkey15 primary key(regno,courseno,sem),

constraint fkey11 foreign key(regno) references student(regno),

constraint fkey13 foreign key(isbn) references text(isbn),

constraint fkey12 foreign key(courseno) references course(courseno)

);

Table created.

SQL> create table book\_adoption

(

courseno int,

sem int,

isbn int,

constraint pkey100 primary key(courseno,sem),

constraint fk11 foreign key(courseno) references course(courseno)

constraint fkey13 foreign key(isbn) references text(isbn),

);

Table created.

* Enter at least five tuples for each relation

SQL> desc student;

Name Null? Type

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

REGNO NOT NULL VARCHAR2(10)

NAME NOT NULL VARCHAR2(10)

MAJOR NOT NULL VARCHAR2(10)

BDATE DATE

SQL> insert into student values('1BI02CS010','Karan','CSE','02-Jan-1984');

1 row created.

SQL> insert into student values('1BI02EE015','Jack','EEE','15-Apr-1983');

1 row created.

SQL> insert into student values('1BI00CS010','Adi','CSE','02-Jan-1982');

1 row created.

SQL> insert into student values('1BI01EC089','Rahul','ECE','01-Dec-1983');

1 row created.

SQL> insert into student values ('1BI01ME075','Sachin','MECH','18-Jul-1983');

1 row created.

SQL> select \* from student;

REGNO NAME MAJOR BDATE

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

1BI01ME075 Sachin MECH 18-JUL-83

1BI02CS010 Karan CSE 02-JAN-84

1BI02EE015 Jack EEE 15-APR-83

1BI00CS010 Adi CSE 02-JAN-82

1BI01EC089 Rahul ECE 01-DEC-83

SQL> desc course;

Name Null? Type

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

COURSENO NOT NULL NUMBER(38)

CNAME NOT NULL VARCHAR2(10)

DEPT NOT NULL VARCHAR2(10)

SQL> insert into course values(11,'DSC','CSE');

1 row created.

SQL> insert into course values(22,'ADA','CSE');

1 row created.

SQL> insert into course values(33,'CN','EC');

1 row created.

SQL> insert into course values(44,'TD','MECH');

1 row created.

SQL> insert into course values(55,'MP','EC');

1 row created.

SQL> select \* from course;

COURSENO CNAME DEPT

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

11 DSC CSE

22 ADA CSE

33 CN EC

44 TD MECH

55 MP EC

SQL> desc enroll;

Name Null? Type

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

REGNO NOT NULL VARCHAR2(10)

COURSENO NOT NULL NUMBER(38)

SEM NOT NULL NUMBER(38)

MARKS NUMBER(38)

SQL> insert into enroll values('1BI02CS010',22,5,72);

1 row created.

SQL> insert into enroll values('1BI00CS010',11,3,90);

1 row created.

SQL> insert into enroll values('1BI01EC089',33,6,52);

1 row created.

SQL> insert into enroll values('1BI01ME075',44,4,85);

1 row created.

SQL> insert into enroll values('1BI02EE015',22,5,75);

1 row created.

SQL> select \* from enroll;

REGNO COURSENO SEM MARKS

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

1BI02CS010 22 5 72

1BI00CS010 11 3 90

1BI01EC089 33 6 52

1BI01ME075 44 4 85

1BI02EE015 22 5 75

SQL> desc text;

Name Null? Type

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

ISBN NOT NULL NUMBER(38)

BOOKTITLE NOT NULL VARCHAR2(20)

PUBLISHER VARCHAR2(20)

AUTHOR VARCHAR2(15)

SQL> insert into text values(7722,'VB6','Dreamtech','Holzner');

1 row created.

SQL> insert into text values(1144,'DS with C','Sapna','Nandagopalan');

1 row created.

SQL> insert into text values(4400,'C Programming','TMH','Balaguruswamy');

1 row created.

SQL> insert into text values(5566,'Computer Nw','PHI','Tennenbaum');

1 row created.

SQL> insert into text values(3388,'MP','PHI','Brey');

1 row created.

SQL> select \* from text;

ISBN BOOKTITLE PUBLISHER AUTHOR

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

7722 VB6 Dreamtech Holzner

1144 DS with C Sapna Nandagopalan

4400 C Programming TMH Balaguruswamy

5566 Computer Nw PHI Tennenbaum

3388 MP PHI Brey

SQL> desc book\_adoption;

Name Null? Type

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

COURSENO NOT NULL NUMBER (38)

SEM NOT NULL NUMBER (38)

ISBN NUMBER (38)

SQL> insert into book\_adoption values(11,3,7722);

1 row created.

SQL> insert into book\_adoption values(22,4,7722);

1 row created.

SQL> insert into book\_adoption values(11,5,4400);

1 row created.

SQL> insert into book\_adoption values(11,8,5566);

1 row created.

SQL> insert into book\_adoption values(55,4,3388);

1 row created.

SQL> insert into book\_adoption values(44,4,5566);

1 row created.

SQL> insert into book\_adoption values(44,7,3388);

1 row created.

SQL> select \* from book\_adoption;

COURSENO SEM ISBN

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

11 3 7722

22 4 7722

11 5 4400

11 8 5566

55 4 3388

44 4 5566

44 7 3388

7 rows selected.

* Demonstrate how you add a new text book to the database and make this book be adopted by some department.

SQL> insert into text values(1234,'Elec.Circuits','Sapna','Giridhar');

1 row created.

SQL> insert into book\_adoption values(55,3,1234);

1 row created.

SQL> select \* from text;

ISBN BOOKTITLE PUBLISHER AUTHOR

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

7722 VB6 Dreamtech Holzner

1144 DS with C Sapna Nandagopalan

4400 C Programming TMH Balaguruswamy

5566 Computer Nw PHI Tennenbaum

3388 MP PHI Brey

1234 Elec.Circuits Sapna Giridhar

6 rows selected.

SQL> select \* from book\_adoption;

COURSENO SEM ISBN

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

11 3 7722

22 4 7722

11 5 4400

11 8 5566

55 4 3388

44 4 5566

44 7 3388

55 3 1234

8 rows selected.

* Produce a list of text books (include course#, book-ISBN, book-title) in the alphabetical order for courses offered by the ‘CS’ department that use more than two books.

SQL> select C.Courseno,T.ISBN,T.Booktitle

from Course C,Book\_adoption BA,Text T

where C.Courseno=BA.Courseno and BA.ISBN=T.ISBN and C.Dept='CSE'

group by C.Courseno,T.ISBN,T.Booktitle;

**OUTPUT:**

COURSENO ISBN BOOKTITLE

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

11 4400 C Programming

11 5566 Computer Nw

11 7722 VB6

22 7722 VB6

* List any department that has all its adopted books published by a specific publisher.

SQL> select distinct c.dept

from course c,bookadoption b,text t

where c.courseno=b.courseno and b.isbn=t.isbn

and not exists (

(select y.bookisbn

from course x,bookadoption y

where x.courseno=y.courseno where x.dept=c.dept)

minus

(select bookisbn from text where publisher='&pub'));

**OUTPUT:**

DEPT

----------

MECH

**EXPERIMENT 4:**

4. The following tables are maintained by a book dealer.

AUTHOR (author-id:int, name:string, city:string, country:string)

PUBLISHER (publisher-id:int, name:string, city:string,country:string)

CATALOG (book-id:int, title:string, author-id:int, publisher-id:int, category-id:int, year:int, price:int)

CATEGORY (category-id:int, description:string)

ORDER-DETAILS (order-no:int, book-id:int, quantity:int)

i. Create the above tables by properly specifying the primary keys and the foreign keys.

ii. Enter at least five tuples for each relation.

iii. Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than the average price of the books in the catalog and the year of

publication is after 2000.

iv. Find the author of the book which has maximum sales.

v. Demonstrate how you increase the price of books published

by a specific publisher by 10%.

vi. Generate suitable reports.

vii. Create suitable front end for querying and displaying the

results.

**RELATIONAL SCHEMA DIAGRAM**

**AUTHOR:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Author-id** | **Name** | **City** | **Country** |

**PUBLISHER**

|  |  |  |  |
| --- | --- | --- | --- |
| **Publisher-id** | **Name** | **City** | **Country** |

**CATALOG:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Book-id** | **Title** | **Author-id** | **Publisher-id** | **Category-id** | **Year** | **Price** |

**CATEGORY:**

|  |  |
| --- | --- |
| **Category-id** | **Description** |

**ORDER-DETAILS:**

|  |  |  |
| --- | --- | --- |
| **Order-no** | **Book-id** | **Quantity** |
|  |  |  |

[ I ] Create the above tables by properly specifying the

primary keys and the foreign keys.

SQL> create table author

(

authorid int,

aname varchar(20),

acity varchar(20),

acountry varchar(20),

constraint pkey16 primary key(authorid)

);

Table created.

SQL> create table publisher

(

publisherid int,

pname varchar(15),

pcity varchar(20),

pcountry varchar(20),

constraint pkey17 primary key(publisherid)

);

Table created.

SQL> create table category

(

categoryid int,

description varchar(20),

constraint pkey primary key(categoryid)

);

Table created.

SQL> create table catalog

(

bookid int,

title varchar(20),

authorid int,

publisherid int,

categoryid int,

year int,

price int,

constraint pkey18 primary key(bookid),

constraint fkey13 foreign key(authorid) references author(authorid),

constraint fkey14 foreign key(publisherid) references publisher(publisherid),

constraint fkey15 foreign key(categoryid) references category(categoryid) );

Table created.

SQL> create table orderdetails

(

orderno int,

bookid int,

quantity int,

constraint pkey19 primary key(orderno,bookid),

constraint fkey16 foreign key(bookid) references catalog(bookid)

);

Table created.

[II] Enter at least five tuples for each relation

SQL> desc author;

Name Null? Type

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

AUTHORID NOT NULL NUMBER(38)

ANAME VARCHAR2(15)

ACITY VARCHAR2(15)

ACOUNTRY VARCHAR2(15)

SQL> insert into Author values(1000,'Nandagopalan','Bangalore','India');

1 row created.

SQL> insert into Author values(2000,'Tony','Haywood','USA');

1 row created.

SQL> insert into Author values(3000,'Holzner','New York','USA');

1 row created.

SQL> insert into Author values(4000,'Tennenbaum','London','UK');

1 row created.

SQL> insert into Author values(5000,'Balaguruswamy','Chennai','India');

1 row created.

SQL> select \* from Author;

AUTHORID ANAME ACITY ACOUNTRY

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

1000 Nandagopalan Bangalore India

2000 Tony Haywood USA

3000 Holzner New York USA

4000 Tennenbaum London UK

5000 Balaguruswamy Chennai India

SQL> desc publisher;

Name Null? Type

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

PUBLISHERID NOT NULL NUMBER(38)

PNAME VARCHAR2(15)

PCITY VARCHAR2(15)

PCOUNTRY VARCHAR2(15)

SQL> insert into publisher values(11,'Wiely','NewDelhi','India');

1 row created.

SQL> insert into publisher values(22,'PHI','California','USA');

1 row created.

SQL> insert into publisher values(33,'Sapna','Bangalore','India');

1 row created.

SQL> insert into publisher values(44,'TMH','NewYork','USA');

1 row created.

SQL> insert into publisher values(55,'Wrox','Texas','USA');

1 row created.

SQL> select \* from publisher;

PUBLISHERID PNAME PCITY PCOUNTRY

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

11 Wiely NewDelhi India

22 PHI California USA

33 Sapna Bangalore India

44 TMH NewYork USA

55 Wrox Texas USA

SQL> desc category;

Name Null? Type

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

CATEGORYID NOT NULL NUMBER(38)

DESCRIPTION VARCHAR2(20)

SQL> insert into category values(1,'OS');

1 row created.

SQL> insert into category values(2,'Languages');

1 row created.

SQL> insert into category values(3,'Hardware');

1 row created.

SQL> insert into category values(4,'Algorithms');

1 row created.

SQL> insert into category values(5,'Internet');

1 row created.

SQL> select \* from category;

CATEGORYID DESCRIPTION

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

1 OS

2 Languages

3 Hardware

4 Algorithms

5 Internet

SQL> desc catalog;

Name Null? Type

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

BOOKID NOT NULL NUMBER(38)

TITLE VARCHAR2(20)

AUTHORID NUMBER(38)

PUBLISHERID NUMBER(38)

CATEGORYID NUMBER(38)

YEAR NUMBER(38)

PRICE NUMBER(38)

SQL> insert into catalog values(123,'DSC',1000,33,2,2000,185);

1 row created.

SQL> insert into catalog values(456,'Networks',4000,44,4,2002,365);

1 row created.

SQL> insert into catalog values(789,'VB6',2000,11,2,2000,300);

1 row created.

SQL> insert into catalog values(213,'Frontpage',4000,44,5,2003,500);

1 row created.

SQL> insert into catalog values(879,'ADA',1000,33,4,2001,195);

1 row created.

SQL> select \* from catalog;

BOOKID TITLE AUTHORID PUBLISHERID CATEGORYID YEAR PRICE

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

123 DSC 1000 33 2 2000 185

456 Networks 4000 44 4 2002 365

789 VB6 2000 11 2 2000 300

213 Frontpage 4000 44 5 2003 500

879 ADA 1000 33 4 2001 195

SQL> desc order\_details;

Name Null? Type

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

ORDERNO NOT NULL NUMBER(38)

BOOKID NOT NULL NUMBER(38)

QUANTITY NUMBER(38)

SQL> insert into order\_details values(112,123,100);

1 row created.

SQL> insert into order\_details values(113,123,20);

1 row created.

SQL> insert into order\_details values(114,213,50);

1 row created.

SQL> insert into order\_details values(115,789,500);

1 row created.

SQL> insert into order\_details values(116,879,8);

1 row created.

SQL> select \* from order\_details;

ORDERNO BOOKID QUANTITY

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

112 123 100

113 123 20

114 213 50

115 789 500

116 879 8

[ I I I ] Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than the average price of the books in the catalog and the year of publication is after 2000.

SQL> select C.Authorid,A.name

from Catalog C,Author A

where A.Authorid=C.Authorid and C.Year>2000 and C.Price >

(Select Avg(Price) from Catalog)

group by C.Authorid,A.name

having count(C.Authorid)>=2;

AUTHORID ANAME

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

4000 Tennenbaum

[ I V ] Find the author of the book which has maximum sales.

SQL> create view salesdetails as(

Select OD.Bookid as Book#,C.Price as Cost,Sum(OD.quantity) as Qty,

sum(OD.quantity\*C.price) as sales

from Order\_details OD,Catalog C,Author A

where OD.Bookid=C.Bookid and C.Authorid=A.Authorid

group by OD.Bookid,C.Price);

View created.

SQL> select A.Authorid,A.Aname,S.Book#,S.Sales

from Author A,Catalog C,Salesdetails S

where A.Authorid=C.Authorid and S.Book#=C.Bookid and sales=(

select Max(Sales) from Salesdetails);

AUTHORID ANAME BOOK# SALES

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

2000 Tony 789 150000

[ V ] Demonstrate how you increase the price of books published by a specific publisher by 10%.

**BEFORE:**

SQL> select \* from Catalog;

BOOKID TITLE AUTHORID PUBLISHERID CATEGORYID YEAR PRICE

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

123 DSC 1000 33 2 2000 185

456 Networks 4000 44 4 2002 365

789 VB6 2000 11 2 2000 300

213 Frontpage 4000 44 5 2003 500

879 ADA 1000 33 4 2001 195

SQL> update catalog

set price=price\*1.10

where publisherid=33;

2 rows updated.

**AFTER:**

SQL> select \* from catalog;

BOOKID TITLE AUTHORID PUBLISHERID CATEGORYID YEAR PRICE

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

123 DSC 1000 33 2 2000 204

456 Networks 4000 44 4 2002 365

789 VB6 2000 11 2 2000 300

213 Frontpage 4000 44 5 2003 500

879 ADA 1000 33 4 2001 215

**EXPERIMENT 5:**

5. Consider the following database for a banking enterprise

BRANCH(branch-name:string, branch-city:string, assets:real)

ACCOUNT(accno:int, branch-name:string, balance:real)

DEPOSITOR(customer-name:string, accno:int)

CUSTOMER(customer-name:string, customer-street:string,

customer-city:string)

LOAN(loan-number:int, branch-name:string, amount:real)

BORROWER(customer-name:string, loan-number:int)

i. Create the above tables by properly specifying the primary keys and the foreign keys

ii. Enter at least five tuples for each relation

iii. Find all the customers who have at least two accounts at the *Main* branch.

iv. Find all the customers who have an account at *all* the branches located in a specific city.

v. Demonstrate how you delete all account tuples at every branch located in a specific city.

vi. Generate suitable reports.

vii. Create suitable front end for querying and displaying the results.

**RELATIONAL SCHEMA DIAGRAM**

**BRANCH:**

|  |  |  |
| --- | --- | --- |
| **Brancname** | **Branch-city** | **Assets** |

**ACCOUNT:**

|  |  |  |
| --- | --- | --- |
| **Accno** | **Branch-name** | **Balance** |

**DEPOSITOR:**

|  |  |
| --- | --- |
| **Customer-name** | **Accno** |

**CUSTOMER:**

|  |  |  |
| --- | --- | --- |
| **Customer-name** | **Customer-street** | **City** |

**LOAN:**

|  |  |  |
| --- | --- | --- |
| **Loan-number** | **Branch-name** | **Amount** |

**BORROWER:**

|  |  |
| --- | --- |
| **Customer-name** | **Uloan-number** |

[ I ] Create the above tables by properly specifying the primary keys and the foreign keys.

SQL> create table branch

(

branchname varchar(20),

branchcity varchar(20),

assets real,

constraint pkey20 primary key(branchname)

);

Table created.

SQL> create table account

(

accno int,

branchname varchar(20),

balance real,

constraint pkey21 primary key(accno)

);

Table created.

SQL> create table customer

(

customername varchar(20),

customerstreet varchar(20),

customercity varchar(20), constraint pkey22 primary key(customername));

Table created.

SQL> create table depositor

(

customername varchar(20),

accno int,

constraint pkey23 primary key(customername,accno),

constraint fkey20 foreign key(customername) references customer(customername),

constraint fkey21 foreign key(accno) references account(accno)on delete cascade

);

Table created.

SQL> create table loan

(

loanno int,

branchname varchar(20),

amount real,

constraint pkey24 primary key(loanno),

constraint fkey22 foreign key(branchname) references branch(branchname)

);

Table created.

SQL> create table borrower

(

customername varchar(20),

loanno int,

constraint pkey25 primary key(customername,loanno),

constraint fkey23 foreign key(customername) references customer(customername),

constraint fkey24 foreign key(loanno) references loan(loanno)

);

Table created.

[II] Enter at least five tuples for each relation

SQL> desc branch;

Name Null? Type

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

BRANCHNAME NOT NULL VARCHAR2 (15)

BRANCHCITY VARCHAR2 (15)

ASSETS NUMBER (63)

SQL> insert into branch values('Jayanagar','Bangalore','15000000');

1 row created.

SQL> insert into branch values('Basavanagudi','Bangalore','25000000');

1 row created.

SQL> insert into branch values('Noida','NewDelhi','50000000');

1 row created.

SQL> insert into branch values('Marinedrive','Mumbai','40000000');

1 row created.

SQL> insert into branch values('GreenPark','Newdelhi','30000000');

1 row created.

SQL> select \* from branch;

BRANCHNAME BRANCHCITY ASSETS

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

Jayanagar Bangalore 15000000

Basavanagudi Bangalore 25000000

Noida NewDelhi 50000000

Marinedrive Mumbai 40000000

GreenPark Newdelhi 30000000

SQL> desc account;

Name Null? Type

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

ACCNO NOT NULL NUMBER(38)

BRANCHNAME VARCHAR2(15)

BALANCE NUMBER(63)

SQL> insert into account values('123','Jayanagar','25000');

1 row created.

SQL> insert into account values('156','Jayanagar','30000');

1 row created.

SQL> insert into account values('456','Basavanagudi','15000');

1 row created.

SQL> insert into account values('789','Noida','25000');

1 row created.

SQL> insert into account values('478','Marinedrive','48000');

1 row created.

SQL> insert into account values('778','GreenPark','60000');

1 row created.

SQL> insert into account values('189','Basavanagudi','48888');

1 row created.

SQL> select \* from account;

ACCNO BRANCHNAME BALANCE

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

123 Jayanagar 25000

156 Jayanagar 30000

456 Basavanagudi 15000

789 Noida 25000

478 Marinedrive 48000

778 GreenPark 60000

189 Basavanagudi 48888

7 rows selected.

SQL> desc customer;

Name Null? Type

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

CUSTOMERNAME NOT NULL VARCHAR2(15)

CUSTOMERSTREET VARCHAR2(15)

CUSTOMERCITY VARCHAR2(15)

SQL> insert into customer values('Ramu','Jayanagar','Bangalore');

1 row created.

SQL> insert into customer values('Kumar','Basavanagudi','Bangalore');

1 row created.

SQL> insert into customer values('John','Noida','Newdelhi');

1 row created.

SQL> insert into customer values('Mike','Marinedrive','Mumbai');

1 row created.

SQL> insert into customer values('Sachin','GreenPark','NewDelhi');

1 row created.

SQL> select \* from customer;

CUSTOMERNAME CUSTOMERSTREET CUSTOMERCITY

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

Ramu Jayanagar Bangalore

Kumar Basavanagudi Bangalore

John Noida Newdelhi

Mike Marinedrive Mumbai

Sachin GreenPark NewDelhi

SQL> desc depositor;

Name Null? Type

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

CUSTOMERNAME NOT NULL VARCHAR2(15)

ACCNO NOT NULL NUMBER(38)

SQL> insert into depositor values('Ramu',123);

1 row created.

SQL> insert into depositor values('Ramu',156);

1 row created.

SQL> insert into depositor values('Ramu',189);

1 row created.

SQL> insert into depositor values('Kumar',456);

1 row created.

SQL> insert into depositor values('John',789);

1 row created.

SQL> insert into depositor values('Mike',478);

1 row created.

SQL> insert into depositor values('Sachin',778);

1 row created.

SQL> select \* from depositor;

CUSTOMERNAME ACCNO

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

Ramu 123

Ramu 156

Ramu 189

Kumar 456

John 789

Mike 478

Sachin 778

7 rows selected.

SQL> desc loan;

Name Null? Type

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

LOANNO NOT NULL NUMBER(38)

BRANCHNAME VARCHAR2(15)

AMOUNT NUMBER(63)

SQL> insert into loan values('1111','Jayanagar','250000');

1 row created.

SQL> insert into loan values('2222','Basavanagudi','350000');

1 row created.

SQL> insert into loan values('3333','Noida','150000');

1 row created.

SQL> insert into loan values('4444','Marinedrive','1500000');

1 row created.

SQL> insert into loan values('5555','GreenPark','7500000');

1 row created.

SQL> select \* from loan;

LOANNO BRANCHNAME AMOUNT

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

1111 Jayanagar 250000

2222 Basavanagudi 350000

3333 Noida 150000

4444 Marinedrive 1500000

5555 GreenPark 7500000

SQL> desc borrower;

Name Null? Type

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

CUSTOMERNAME NOT NULL VARCHAR2(15)

LOANNO NOT NULL NUMBER(38)

SQL> insert into borrower values('Ramu',1111);

1 row created.

SQL> insert into borrower values('Kumar',2222);

1 row created.

SQL> insert into borrower values('John',3333);

1 row created.

SQL> insert into borrower values('Mike',4444);

1 row created.

SQL> insert into borrower values('Sachin',5555);

1 row created.

SQL> select \* from borrower;

CUSTOMERNAME LOANNO

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

Ramu 1111

Kumar 2222

John 3333

Mike 4444

Sachin 5555

[ III ] Find all the customers who have atleast 2 accounts at main branch.

SQL> SELECT C.customername

FROM customer C,depositor D,account A

WHERE C.customername=D.customername AND D.accno=A.accno AND

A.branchname = 'Jayanagar'

group by C.customername HAVING COUNT(A.accno)>=2;

CUSTOMERNAME

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

Ramu

[ I V ] Find all the customers who have an account at all the branches located in a specific city.

SQL> SELECT \* FROM account A,depositor D,bcustomer C,branch B

WHERE C.cname=D.cname AND D.accno=A.accno AND

A.bname=B.bname AND B.bcity='MYS'

GROUP BY C.cname

HAVING COUNT(distinct B.bname)=(SELECT COUNT(bname) FROM branch

WHERE bcity='MYS');

CUSTOMERNAME

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

Ramu

[V] Demonstrate how you delete all account tuples at every branch located in a specific city.

DELETE FROM account

WHERE branchname IN (SELECT branchname FROM branch WHERE branchcity='Bangalore');

**SAMPLE VIVA QUESTION**

* What is Database Management System (DBMS)?
* What are the benefits of using DBMS?
* What are the different types of databases?
* Who are the different types of database users? What are their role and responsibilities?
* What are the advantages of using Database system over File System?
* What are the different applications of DBMS?
* What is SQL?
* Explain ER diagram for all the relational tables used in the lab manual.
* What are the differences between Database and Relational Database?
* Explain CREATE, INSERT, SELECT, UPDATE, DELETE, DROP, ALTER commands used in SQL.
* What are the different data types used in SQL-99?
* How VARCHAR(*Size*) is better than CHAR(*Size*)?
* What are the different Integrity Constraints? Or Explain different types of keys used in SQL?
* Why we need to create and join multiple tables?
* What are the different types of JOIN used in SQL?
* Explain EQUIE JOIN and the significance of creating mirror copy? Why it is called PIG EAR JOIN?
* How to implement simple join? How it is Cartesian product of multiple tables?
* What is the difference between DELETE and DROP commands?
* How would you add or drop constraints from table give on example.
* How to rename a table?
* What is the usage of creating views?
* Under what circumstances a VIEW cannot be updated?
* Why we are using Normalization?
* What are the different types of Normalization?
* Which normalization is based on atomic property of attributes in a table?
* What are the normalizations based on Functional Dependencies of attributes in a table?
* Write relational algebra and calculus for any given queries.
* Why we are using GROUP BY HAVING clause in SQL?
* What are the different Aggregation functions in SQL?
* What do you mean by embedded SQL?
* What are assertions and triggers?
* What do you mean by transaction
* What are the different properties of a transaction?
* What are the methods of doing concurrency control?
* What are the different types of LOCKS?