AISHWARYA M 1BM20CS401 CSE-4A

Program 6: Order Database

Consider the following schema for Order Database:

```
SALESMAN (Salesman_id, Name, City, Commission)
CUSTOMER (Customer_id, Cust_Name, City, Grade,
Salesman_id)
ORDERS (Ord_No, Purchase_Amt, Ord_Date, Customer_id,
Salesman_id)
Write SQL queries to
```

- 1. Count the customers with grades above Bangalore's average.
- 2. Find the name and numbers of all salesmen who had more than one customer.
- 3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)
- 4. Create a view that finds the salesman who has the customer with the highest order of a day.
- 5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

create database Orderdb; use Orderdb;

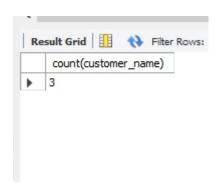
```
create table Salesman(
salesman_id int not null,
salesman name varchar(20) not null,
city varchar(20) not null,
commission int not null,
primary key(salesman_ id)
);
create table Customer(
customer id int not null,
customer name varchar(20) not null,
city varchar(20) not null,
grade int not null,
salesman id int,
primary key(customer_id),
foreign key(salesman id)references
Salesman (salesman id) on delete set null
);
create table Orders(
order_id int not null,
purchase_amt int not null,
order date date not null,
customer id int not null,
salesman id int,
```

primary key(order_id),

```
foreign key(customer id)references
Customer(customer id),
foreign key(salesman_id)references
Salesman (salesman id) on delete set null
);
insert into Salesman
 values(1000, 'John', 'Bangalore', 25),
 (2000, 'Ravi', 'Bangalore', 20),
 (3000, 'Kumar', 'Mysore', 15),
 (4000, 'Smith', 'Delhi', 30),
 (5000, 'Harsha', 'Hyderabad', 15);
 insert into Customer
 values(10,'Preethi','Bangalore',100,1000),
 (11,'Vivek','Mangalore',300,1000),
 (12, 'Bhaskar', 'Chennai', 400, 2000),
 (13,'Chethan','Bangalore',200,2000),
 (14, 'Mamatha', 'Bangalore', 400, 3000);
 insert into Orders
 values(50,5000,'2017-05-04',10,1000),
 (51,450,'2017-01-20',10,2000),
 (52,1000,'2017-02-24',13,2000),
 (53,3500,'2017-04-13',14,3000),
 (54,550,'2017-03-09',12,2000);
```

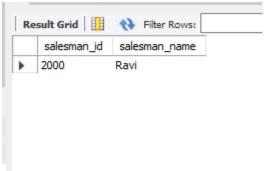
---- Count the customers with grades above Bangalore's average.

select count(customer_name)from Customer
where grade> (Select avg(grade) from Customer
where city ='Bangalore');



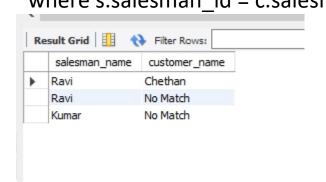
----- Find the name and numbers of all salesmen who had more than one customer.

select distinct c.salesman_id,s.salesman_name
from Customer c,Salesman s
 where c.salesman_id=s.salesman_id
 and 1<(select count(customer_id) from Customer
 where salesman_id=c.salesman_id);</pre>



---- List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

select s.salesman_name,c.customer_name from
Salesman s,Customer c
where s.salesman_id=c.salesman_id and c.city=s.city
union
select s.salesman_name,'No Match' from Salesman
s,Customer c
where s.salesman_id = c.salesman_id and c.city!=s.city;



-----Create a view that finds the salesman who has the customer with the highest order of a day.

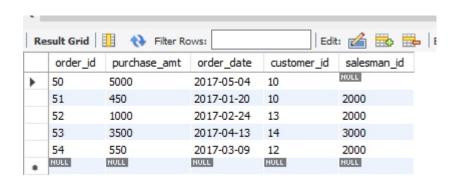
create view salesman_view as select
o.order_date ,salesman_id,sum(o.purchase_amt)from
Orders o group by order_date
having sum(purchase_amt)=(select
max(sum(purchase_amt))from Customer
where order_date =o.order_date and salesman_id
=o.salesman_id);

	order_date	salesman_id	sum(o.purchase_amt)
•	2017-01-20	2000	450
	2017-02-24	2000	1000
	2017-04-13	3000	3500
	2017-03-09	2000	550

-----Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

delete from Salesman where salesman_id=1000; select*from Salesman; select * from Orders;





Program 7: Book Database

```
BOOK (Book_id, Title, Publisher_Name, Pub_Year)
BOOK_AUTHORS (Book_id, Author_Name)
PUBLISHER (Name, Address, Phone)
BOOK_COPIES (Book_id, Branch_id, No-of_Copies)
BOOK_LENDING (Book_id, Branch_id, Card_No, Date_Out, Due_Date)
LIBRARY BRANCH (Branch_id, Branch_Name, Address)
```

Write SQL queries to

- 1. Retrieve details of all books in the library id, title, name of publisher, authors, number of copies in each branch, etc.
- 2. Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017
- 3. Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.
- 4. Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.
- 5. Create a view of all books and its number of copies that are currently available in the Library.

```
use bookdb;

create table publisher(
name varchar(30) not null,
address varchar(20),
phone varchar(10),
primary key(name)
);
```

create database bookdb;

```
create table book(
 book id int not null,
 title varchar(20),
 publisher_name varchar(20),
 pub year varchar(20),
primary key(book_id),
 foreign key(publisher_name) references
publisher(name)
 );
 create table book authors(
 book_id int not null,
 author name varchar(30) not null,
 primary key(book_id,author_name),
 foreign key(book_id) references book(book_id)
 );
 create table library_branch(
 branch id int not null,
 address varchar(20),
 branch_name varchar(20),
 primary key(branch id)
 );
 create table book copies(
 book_id int not null,
 branch id int not null,
 no_of_copies int,
```

```
primary key(book id, branch id),
foreign key(book_id) references book(book_id),
foreign key(branch id) references
library branch(branch id)
);
create table Card(
card_no int(10) not null,
primary key(card_no)
);
create table book_lending(
date out date,
due date date,
book id int
              not null,
branch id int not null,
card no int not null,
primary key(book_id,branch_id,card_no),
foreign key(book_id) references book(book_id),
foreign key(branch id) references
library branch(branch id),
foreign key(card_no) references Card(card_no)
);
insert into publisher
values('MCGRAW-HILL', 'BANGALORE',9989076587),
('PEARSON', 'NEWDELHI', 9889076565),
('RANDOM HOUSE', 'HYDRABAD', 7455679345),
('HACHETTE LIVRE', 'CHENAI', 8970862340),
```

('GRUPO PLANETA', 'BANGALORE', 7756120238);

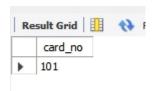
```
insert into book
values(1,'DBMS', 'MCGRAW-HILL','JAN-2017'),
 (2,'ADBMS', 'MCGRAW-HILL','JUN-2016'),
 (3,'CN', 'PEARSON','SEP-2016'),
 (4,'CG', 'GRUPO PLANETA','SEP-2015'),
 (5,'OS', 'PEARSON','MAY-2016');
 insert into book authors
 values(1,'NAVATHE'),
 (2,'NAVATHE'),
 (3,'TANENBAUM'),
 (4, 'EDWARD ANGEL'),
(5,'GALVIN');
insert into library branch
values(10,'RR NAGAR','BANGALORE'),
 (11, 'RNSIT', 'BANGALORE'),
 (12, 'RAJAJI NAGAR', 'BANGALORE'),
 (13,'NITTE','MANGALORE'),
 (14, 'MANIPAL', 'UDUPI');
 insert into book copies
 values(1, 10,10),
 (1, 11, 5),
 (2, 12, 2),
```

```
(2, 13, 5),
 (3, 14, 7),
 (5, 10, 1),
(4, 11, 3);
insert into Card
values(100),
(101),
(102),
(103),
(104);
insert into book lending
values('2017-01-01','2017-06-01', 1, 10, 101),
('2017-01-01','2017-03-11', 3, 14, 101),
('2017-02-21','2017-04-21', 2, 13, 101),
 ('2017-03-15','2017-07-15', 4, 11, 101),
 ('2017-04-12','2017-05-12', 1, 11, 104);
-----Retrieve details of all books in the library – id, title,
name of publisher, authors, number of copies in each
branch, etc.
select
b.book id,b.title,b.publisher name,a.author name,c.no
of copies, l. branch id
from book b,book_authors a,book copies
c, library branch l
where b.book id=a.book id and b.book id=c.book id
and l.branch id=c.branch id;
```



-----Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017

select card_no from book_lending where date_out between '2017-01-01' and '2017-07-01' group by card_no having count(*)>3;



-----Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.

DELETE FROM BOOK WHERE BOOK_ID=3;

select * from book;

	BOOK_ID	TITLE	PUB_YEAR	PUBLISHER_NAME
•	1	DBMS	JAN-2017	MCGRAW-HILL
	2	ADBMS	JUN-2016	MCGRAW-HILL
	4	CG	SEP-2015	GRUPO PLANETA
	5	OS	MAY-2016	PEARSON
	NULL	NULL	NULL	MULL

select * from book_authors;

	AUTHOR_NAME	BOOK_ID
•	NAVATHE	1
	NAVATHE	2
	EDWARD ANGEL	4
	GALVIN	5
	NULL	NULL

select * from book lending;

	DATE_OUT	DUE_DATE	BOOK_ID	BRANCH_ID	CARD_NO
١	2017-01-01	2017-06-01	1	10	101
	2017-04-12	2017-05-12	1	11	104
	2017-02-21	2017-04-21	2	13	101
	2017-01-17	2017-03-17	3	14	101
	2017-03-15	2017-07-15	4	11	101
	NULL	NULL	NULL	NULL	HULL

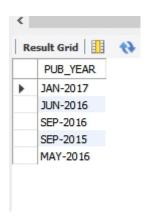
select * from book_copies;

	NO_OF_COPIES	BOOK_ID	BRANCH_ID
•	10	1	10
	5	1	11
	2	2	12
	5	2	13
	3	4	11
	1	5	10
	NULL	NULL	NULL

-----Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.

CREATE VIEW YEAR_OF_PUBLICATION AS SELECT PUB_YEAR FROM BOOK;

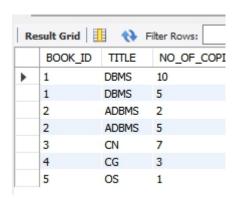
SELECT * FROM YEAR_OF_PUBLICATION;



-----Create a view of all books and its number of copies that are currently available in the Library.

CREATE VIEW BOOKS_AVAILABLE_IN_LIBRARY
AS SELECT B.BOOK_ID, B.TITLE,
C.NO_OF_COPIES FROM BOOK B, BOOK_COPIES
C, LIBRARY_BRANCH L
WHERE B.BOOK_ID=C.BOOK_ID AND
C.BRANCH_ID=L.BRANCH_ID;

SELECT * FROM BOOKS AVAILABLE IN LIBRARY;



Program 8:Student Enrolment Database

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)

Database applications laboratory GCEM DEPARTMENT OF CSE Page - 5 - 5th semester

- 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. 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.

```
CREATE DATABASE COLLEGE;
USE COLLEGE;
CREATE TABLE student(
     regno VARCHAR(15),
     sname VARCHAR(20),
     major VARCHAR(20),
     bdate DATE,
     PRIMARY KEY (regno)
     );
CREATE TABLE course(
     courseno INT,
     cname VARCHAR(20),
     dept VARCHAR(20),
     PRIMARY KEY (courseno)
     );
CREATE TABLE enroll(
     regno VARCHAR(15),
     courseno INT,
     sem INT(3),
     marks INT(4),
     PRIMARY KEY (regno, courseno),
     FOREIGN KEY (regno) REFERENCES student
(regno),
     FOREIGN KEY (courseno) REFERENCES course
(courseno)
     );
CREATE TABLE text(
```

```
book isbn INT(5),
      book title VARCHAR(20),
      publisher VARCHAR(20),
      author VARCHAR(20),
      PRIMARY KEY (book isbn)
      );
CREATE TABLE book adoption(
      courseno INT,
      sem INT(3),
      book isbn INT(5),
      PRIMARY KEY (courseno, book isbn),
      FOREIGN KEY (courseno) REFERENCES course
(courseno),
      FOREIGN KEY (book isbn) REFERENCES
text(book_isbn)
      );
INSERT INTO student
 VALUES('1pe11cs002','b','sr','19930924'),
      ('1pe11cs003','c','sr','19931127'),
      ('1pe11cs004','d','sr','19930413'),
      ('1pe11cs005','e','jr','19940824');
 INSERT INTO student
 VALUES('1pe11cs001', 'a', 'jr', '19930922');
select * from student;
```

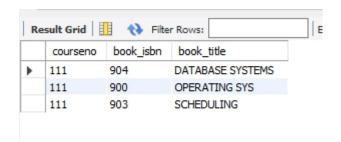
```
INSERT INTO course
 VALUES (111, 'OS', 'CSE'),
          (112, 'EC', 'CSE'),
          (113,'SS','ISE'),
          (114, 'DBMS', 'CSE'),
          (115, 'SIGNALS', 'ECE');
select * from course:
INSERT INTO text
 VALUES (10, DATABASE
 SYSTEMS', 'PEARSON', 'SCHIELD'),
      (900, 'OPERATING SYS', 'PEARSON', 'LELAND'),
      (901, 'CIRCUITS', 'HALL INDIA', 'BOB'),
      (902, 'SYSTEM SOFTWARE', 'PETERSON', 'JACOB'),
      (903, 'SCHEDULING', 'PEARSON', 'PATIL'),
      (904, 'DATABASE SYSTEMS', 'PEARSON', 'JACOB'),
      (905, 'DATABASE MANAGER', 'PEARSON', 'BOB'),
      (906, 'SIGNALS', 'HALL INDIA', 'SUMIT');
select * from text:
INSERT INTO enroll
 VALUES ('1pe11cs001',115,3,100),
      ('1pe11cs002',114,5,100),
      ('1pe11cs003',113,5,100),
      ('1pe11cs004',111,5,100),
      ('1pe11cs005',112,3,100);
select * from enroll;
```

INSERT INTO book adoption

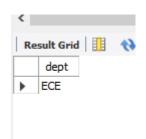
```
VALUES(111,5,900),
(111,5,903),
(111,5,904),
(112,3,901),
(113,3,10),
(114,5,905),
(113,5,902),
(115,3,906);
select * from book_adoption;
```

----- 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.

```
SELECT c.courseno,t.book_isbn,t.book_title
FROM course c,book_adoption ba,text t
WHERE c.courseno=ba.courseno
AND ba.book_isbn=t.book_isbn
AND c.dept='CSE'
AND 2<(
SELECT COUNT(book_isbn)
FROM book_adoption b
WHERE c.courseno=b.courseno)
ORDER BY t.book_title;
```



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



Program 9: Movie database

Consider the schema for Movie Database:

```
ACTOR (Act_id, Act_Name, Act_Gender)
DIRECTOR (Dir_id, Dir_Name, Dir_Phone)
MOVIES (Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)
MOVIE_CAST (Act_id, Mov_id, Role)
RATING (Mov_id, Rev_Stars)
Write SQL queries to
```

- 1. List the titles of all movies directed by 'Hitchcock'.
- 2. Find the movie names where one or more actors acted in two or more movies.
- 3. List all actors who acted in a movie before 2000 and also in a movie after
- 2015 (use JOIN operation).
- 4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.
- 5. Update rating of all movies directed by 'Steven Spielberg' to 5

CREATE DATABASE MOVIE; USE MOVIE;

CREATE TABLE ACTOR (
ACT_ID INT,
ACT_NAME VARCHAR (20),
ACT_GENDER CHAR (1),

```
PRIMARY KEY (ACT_ID));
CREATE TABLE DIRECTOR (
DIR ID INT,
DIR NAME VARCHAR (20),
DIR PHONE LONG,
PRIMARY KEY (DIR_ID));
CREATE TABLE MOVIES (
MOV_ID INT,
MOV TITLE VARCHAR (25),
MOV YEAR INT,
MOV LANG VARCHAR (12),
DIR ID INT,
PRIMARY KEY (MOV ID),
FOREIGN KEY (DIR_ID) REFERENCES DIRECTOR
(DIR ID));
CREATE TABLE MOVIE_CAST (
ACT ID INT,
MOV ID INT,
AROLE VARCHAR(10),
PRIMARY KEY (ACT_ID, MOV_ID),
FOREIGN KEY(ACT_ID) REFERENCES ACTOR(ACT_ID)
ON DELETE CASCADE,
FOREIGN KEY(MOV ID) REFERENCES
MOVIES(MOV ID) ON DELETE CASCADE);
```

```
CREATE TABLE RATING (
MOV ID INT,
REV_STARS VARCHAR (25),
PRIMARY KEY (MOV ID),
FOREIGN KEY (MOV ID) REFERENCES MOVIES
(MOV ID));
INSERT INTO ACTOR
VALUES (301, 'ANUSHKA', 'F'),
 (302, 'PRABHAS', 'M'),
 (303, 'PUNITH', 'M'),
 (304, 'JERMY', 'M');
select * from actor;
INSERT INTO DIRECTOR
 VALUES (60, 'RAJAMOULI', 8751611001),
  (61, 'HITCHCOCK', 7766138911),
  (62, 'FARAN', 9986776531),
  (63, 'STEVEN SPIELBERG', 8989776530);
select * from director;
INSERT INTO MOVIES
VALUES (1001, 'BAHUBALI-2', 2017, 'TELAGU', 60),
 (1002, 'BAHUBALI-1', 2015, 'TELAGU', 60),
 (1003, 'AKASH', 2008, 'KANNADA', 61),
 (1004, 'WAR HORSE', 2011, 'ENGLISH', 63);
select * from movies;
```

```
INSERT INTO MOVIE CAST
VALUES (301, 1002, 'HEROINE'),
 (301, 1001, 'HEROINE'),
 (303, 1003, 'HERO'),
 (303, 1002, 'GUEST'),
 (304, 1004, 'HERO');
select * from movie_cast;
INSERT INTO RATING
VALUES (1001, 4),
 (1002, 2),
 (1003, 5),
 (1004, 4);
select * from rating;
----- List the titles of all movies directed by
'Hitchcock'.
SELECT MOV TITLE
FROM MOVIES
WHERE DIR_ID IN (SELECT DIR_ID
FROM DIRECTOR
WHERE DIR_NAME = 'HITCHCOCK');
```



----- Find the movie names where one or more actors acted in two or more movies.

SELECT MOV_TITLE
FROM MOVIES M, MOVIE_CAST MV
WHERE M.MOV_ID=MV.MOV_ID AND ACT_ID IN
(SELECT ACT_ID
FROM MOVIE_CAST GROUP BY ACT_ID
HAVING COUNT(ACT_ID)>1)
GROUP BY MOV_TITLE
HAVING COUNT(*)>1;



--- List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

SELECT ACT_NAME, MOV_TITLE, MOV_YEAR

FROM ACTOR A

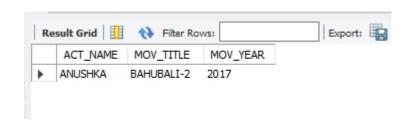
JOIN MOVIE_CAST C

ON A.ACT_ID=C.ACT_ID

JOIN MOVIES M

ON C.MOV_ID=M.MOV_ID

WHERE M.MOV_YEAR NOT BETWEEN 2000 AND 2015;



----- Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

SELECT MOV_TITLE, MAX(REV_STARS)
FROM MOVIES
INNER JOIN RATING USING (MOV_ID)
GROUP BY MOV_TITLE
HAVING MAX(REV_STARS)>0
ORDER BY MOV_TITLE;



----- Update rating of all movies directed by 'Steven Spielberg' to 5 KL.

UPDATE RATING

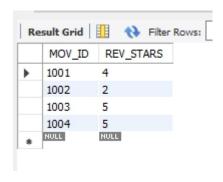
SET REV_STARS=5

WHERE MOV_ID IN(SELECT MOV_ID FROM MOVIES

WHERE DIR_ID IN(SELECT DIR_ID FROM DIRECTOR

WHERE DIR_NAME = 'STEVEN SPIELBERG'));

select * from rating;



Program 10:College Database

Consider the schema for College Database:

STUDENT (USN, SName, Address, Phone, Gender)
SEMSEC (SSID, Sem, Sec)
CLASS (USN, SSID)
SUBJECT (Subcode, Title, Sem, Credits)
IAMARKS (USN, Subcode, SSID, Test1, Test2, Test3, FinalIA)
Write SQL queries to

- 1. List all the student details studying in fourth semester 'C' section.
- 2. Compute the total number of male and female students in each semester and in each section.
- 3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.
- 4. Categorize students based on the following criterion:

If FinalIA = 17 to 20 then CAT = 'Outstanding' If FinalIA = 12 to 16 then CAT = 'Average' If FinalIA < 12 then CAT = 'Weak' Give these details only for 8th semester A, B, and C section students.

CREATE DATABASE COLLEGEDB; USE COLLEGEDB;

CREATE TABLE STUDENT (
USN VARCHAR (10),
SNAME VARCHAR (25),
ADDRESS VARCHAR (25),
PHONE LONG,
GENDER CHAR (1),
PRIMARY KEY (USN));

```
CREATE TABLE SEMSEC (
SSID VARCHAR (5),
SEM INT,
SEC CHAR (1),
PRIMARY KEY (SSID));
CREATE TABLE CLASS (
USN VARCHAR (10),
SSID VARCHAR (5),
PRIMARY KEY (USN, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN),
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));
CREATE TABLE SUBJECT (
SUBCODE VARCHAR (8),
TITLE VARCHAR (20),
SEM INT,
CREDITS INT,
PRIMARY KEY (SUBCODE));
CREATE TABLE IAMARKS (
USN VARCHAR (10),
SUBCODE VARCHAR (8),
SSID VARCHAR (5),
TEST1 INT,
TEST2 INT,
TEST3 INT,
FINALIA INT,
PRIMARY KEY (USN, SUBCODE, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN),
```

```
FOREIGN KEY (SUBCODE) REFERENCES SUBJECT
(SUBCODE),
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));
INSERT INTO STUDENT VALUES
('1RN13CS020','AKSHAY','BELAGAVI', 8877881122,'M');
INSERT INTO STUDENT VALUES
('1RN13CS062', 'SANDHYA', 'BENGALURU',
7722829912,'F');
INSERT INTO STUDENT VALUES
('1RN13CS091','TEESHA','BENGALURU', 7712312312,'F');
INSERT INTO STUDENT VALUES
('1RN13CS066', 'SUPRIYA', 'MANGALURU',
8877881122,'F');
INSERT INTO STUDENT VALUES
('1RN14CS010','ABHAY','BENGALURU', 9900211201,'M');
INSERT INTO STUDENT VALUES
('1RN14CS032', 'BHASKAR', 'BENGALURU',
9923211099,'M'):
INSERT INTO STUDENT VALUES
('1RN14CS025','ASMI','BENGALURU', 7894737377,'F');
INSERT INTO STUDENT VALUES
('1RN15CS011','AJAY','TUMKUR', 9845091341,'M');
INSERT INTO STUDENT VALUES
('1RN15CS029','CHITRA','DAVANGERE', 7696772121,'F');
INSERT INTO STUDENT VALUES
('1RN15CS045','JEEVA','BELLARY', 9944850121,'M');
INSERT INTO STUDENT VALUES
('1RN15CS091','SANTOSH','MANGALURU',
8812332201,'M');
```

```
INSERT INTO STUDENT VALUES
('1RN16CS045','ISMAIL','KALBURGI', 9900232201,'M');
INSERT INTO STUDENT VALUES
('1RN16CS088','SAMEERA','SHIMOGA', 9905542212,'F');
INSERT INTO STUDENT VALUES
('1RN16CS122','VINAYAKA','CHIKAMAGALUR',
8800880011,'M');
Select * from STUDENT;
```

```
INSERT INTO SEMSEC VALUES ('CSE8A', 8,'A');
INSERT INTO SEMSEC VALUES ('CSE8B', 8,'B');
INSERT INTO SEMSEC VALUES ('CSE8C', 8,'C');
INSERT INTO SEMSEC VALUES ('CSE7A', 7,'A');
INSERT INTO SEMSEC VALUES ('CSE7B', 7,'B');
INSERT INTO SEMSEC VALUES ('CSE7C', 7,'C');
INSERT INTO SEMSEC VALUES ('CSE6A', 6,'A');
INSERT INTO SEMSEC VALUES ('CSE6B', 6,'B');
INSERT INTO SEMSEC VALUES ('CSE6C', 6,'C');
INSERT INTO SEMSEC VALUES ('CSE5A', 5,'A');
INSERT INTO SEMSEC VALUES ('CSE5B', 5,'B');
INSERT INTO SEMSEC VALUES ('CSE5C', 5,'C');
INSERT INTO SEMSEC VALUES ('CSE4A', 4,'A');
INSERT INTO SEMSEC VALUES ('CSE4B', 4,'B');
INSERT INTO SEMSEC VALUES ('CSE4C', 4,'C');
INSERT INTO SEMSEC VALUES ('CSE3A', 3,'A');
INSERT INTO SEMSEC VALUES ('CSE3B', 3,'B');
INSERT INTO SEMSEC VALUES ('CSE3C', 3,'C');
INSERT INTO SEMSEC VALUES ('CSE2A', 2,'A');
INSERT INTO SEMSEC VALUES ('CSE2B', 2,'B');
INSERT INTO SEMSEC VALUES ('CSE2C', 2,'C');
```

```
INSERT INTO SEMSEC VALUES ('CSE1A', 1,'A');
INSERT INTO SEMSEC VALUES ('CSE1B', 1,'B');
INSERT INTO SEMSEC VALUES ('CSE1C', 1,'C');
Select * from SEMSEC;
```

```
INSERT INTO CLASS VALUES ('1RN13CS020','CSE8A');
INSERT INTO CLASS VALUES ('1RN13CS062','CSE8A');
INSERT INTO CLASS VALUES ('1RN13CS066','CSE8B');
INSERT INTO CLASS VALUES ('1RN13CS091','CSE8C');
INSERT INTO CLASS VALUES ('1RN14CS010','CSE7A');
INSERT INTO CLASS VALUES ('1RN14CS025','CSE7A');
INSERT INTO CLASS VALUES ('1RN14CS032','CSE7A');
INSERT INTO CLASS VALUES ('1RN15CS011','CSE4A');
INSERT INTO CLASS VALUES ('1RN15CS029','CSE4A');
INSERT INTO CLASS VALUES ('1RN15CS045','CSE4B');
INSERT INTO CLASS VALUES ('1RN15CS091','CSE4C');
INSERT INTO CLASS VALUES ('1RN16CS045','CSE3A');
INSERT INTO CLASS VALUES ('1RN16CS088','CSE3B');
INSERT INTO CLASS VALUES ('1RN16CS122','CSE3C');
Select * from CLASS;
```

INSERT INTO SUBJECT VALUES ('10CS81','ACA', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS82','SSM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS83','NM', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS84','CC', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS85','PW', 8, 4);
INSERT INTO SUBJECT VALUES ('10CS71','OOAD', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS72','ECS', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS73','PTW', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS74','DWDM', 7, 4);
INSERT INTO SUBJECT VALUES ('10CS75','JAVA', 7, 4);

```
INSERT INTO SUBJECT VALUES ('10CS76', 'SAN', 7, 4);
INSERT INTO SUBJECT VALUES ('15CS51', 'ME', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS52','CN', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS53', 'DBMS', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS54', 'ATC', 5, 4);
INSERT INTO SUBJECT VALUES ('15CS55','JAVA', 5, 3);
INSERT INTO SUBJECT VALUES ('15CS56', 'AI', 5, 3);
INSERT INTO SUBJECT VALUES ('15CS41', 'M4', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS42','SE', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS43','DAA', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS44', 'MPMC', 4, 4);
INSERT INTO SUBJECT VALUES ('15CS45','OOC', 4, 3);
INSERT INTO SUBJECT VALUES ('15CS46','DC', 4, 3);
INSERT INTO SUBJECT VALUES ('15CS31', 'M3', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS32', 'ADE', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS33', 'DSA', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS34','CO', 3, 4);
INSERT INTO SUBJECT VALUES ('15CS35','USP', 3, 3);
INSERT INTO SUBJECT VALUES ('15CS36', 'DMS', 3, 3);
Select * from SUBJECT;
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1,
```

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1RN13CS091','10CS81','CSE8C', 15, 16, 18);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1RN13CS091','10CS82','CSE8C', 12, 19, 14);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1RN13CS091','10CS83','CSE8C', 19, 15, 20);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1RN13CS091','10CS84','CSE8C', 20, 16, 19);

INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES ('1RN13CS091','10CS85','CSE8C', 15, 15, 12);

Select * from IAMARKS;

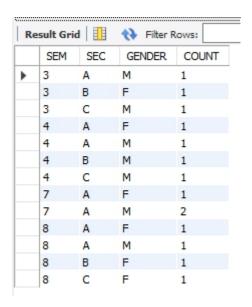
----- List all the student details studying in fourth semester 'C' section.

SELECT S.*, SS.SEM, SS.SEC FROM STUDENT S, SEMSEC SS, CLASS C WHERE S.USN = C.USN AND SS.SSID = C.SSID AND SS.SEM = 4 AND SS.SEC='C';



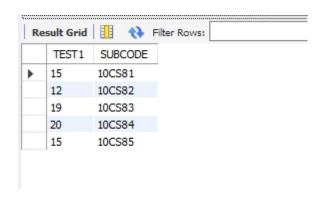
---- Compute the total number of male and female students in each semester and in each section.

SELECT SS.SEM, SS.SEC, S.GENDER, COUNT(S.GENDER)
AS COUNT
FROM STUDENT S, SEMSEC SS, CLASS C
WHERE S.USN = C.USN AND
SS.SSID = C.SSID
GROUP BY SS.SEM, SS.SEC, S.GENDER
ORDER BY SEM;



----- Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

CREATE VIEW STU_TEST1_MARKS_VIEW
AS
SELECT TEST1, SUBCODE
FROM IAMARKS
WHERE USN = '1RN13CS091';



----- Categorize students based on the following criterion:

If FinalIA = 17 to 20 then CAT = 'Outstanding'

If FinalIA = 12 to 16 then CAT = 'Average'

If FinalIA< 12 then CAT = 'Weak'

Give these details only for 8th semester A, B, and C section students.

