1. 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.

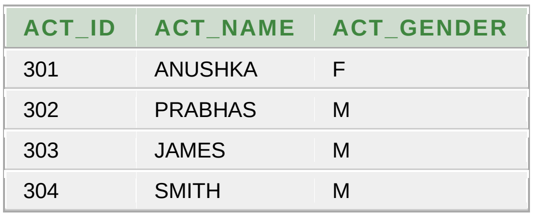
3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

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

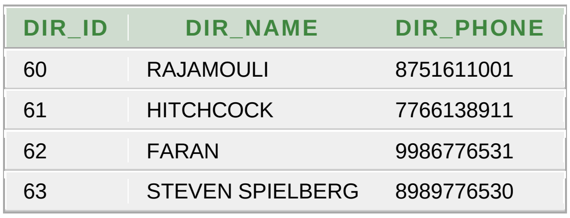
```
create table actor(
act_id number(3) primary key,
act_name varchar(50),
act_gender char
);


create table director(
dir_id number(3) primary key,
dir_name varchar(50),
dir_phone number(10)
);


create table movies(
mov_id number(3) primary key,
mov_title varchar(40),
mov_year number(4),
mov_lang varchar(10),
dir_id number(3),
foreign key (dir_id) references director(dir_id)
);
```

```
create table movie_cast(
act_id number(3),
mov_id number(3),
role varchar(15),
foreign key (act_id) references actor (act_id),
foreign key (mov_id) references movies (mov_id)
);

create table rating(
mov_id number(3),
rev_stars float,
foreign key (mov_id) references movies(mov_id)
);

INSERT INTO actor VALUES (301,'ANUSHKA','F');
INSERT INTO actor VALUES (302,'PRABHAS','M');
INSERT INTO actor VALUES (303,'JAMES','M');
INSERT INTO actor VALUES (304,'SMITH','M');
SELECT * FROM actor;
```



ACT_ID	ACT_NAME	ACT_GENDER
301	ANUSHKA	F
302	PRABHAS	M
303	JAMES	M
304	SMITH	M

```
INSERT INTO director VALUES (60,'RAJAMOULI', 8751611001);
INSERT INTO director VALUES (61,'HITCHCOCK', 7766138911);
INSERT INTO director VALUES (62,'FARAN', 9986776531);
 INSERT INTO director VALUES (63,'STEVEN SPIELBERG', 8989776530);
SELECT * FROM director;
```



DIR_ID	DIR_NAME	DIR_PHONE
60	RAJAMOULI	8751611001
61	HITCHCOCK	7766138911
62	FARAN	9986776531
63	STEVEN SPIELBERG	8989776530

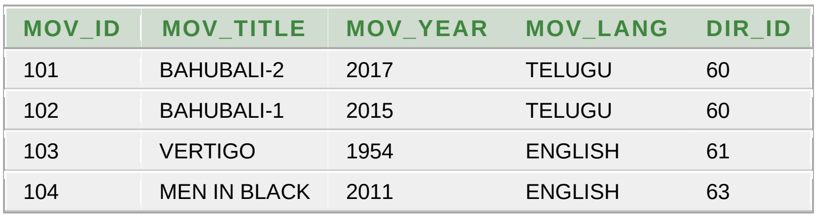
INSERT INTO movies VALUES (101,'BAHUBALI-2', 2017, 'TELUGU', 60);

INSERT INTO movies VALUES (102,'BAHUBALI-1', 2015, 'TELUGU', 60);

INSERT INTO movies VALUES (103,'VERTIGO', 1954, 'ENGLISH', 61);

INSERT INTO movies VALUES (104,'MEN IN BLACK', 2011, 'ENGLISH', 63);

SELECT * FROM movies;



MOV_ID	MOV_TITLE	MOV_YEAR	MOV_LANG	DIR_ID
101	BAHUBALI-2	2017	TELUGU	60
102	BAHUBALI-1	2015	TELUGU	60
103	VERTIGO	1954	ENGLISH	61
104	MEN IN BLACK	2011	ENGLISH	63

INSERT INTO movie_cast VALUES (301, 102, 'HEROINE');

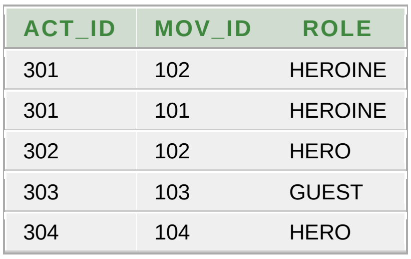
INSERT INTO movie_cast VALUES (301, 101, 'HEROINE');

INSERT INTO movie_cast VALUES (302, 102, 'HERO');

INSERT INTO movie_cast VALUES (303, 103, 'GUEST');

INSERT INTO movie_cast VALUES (304, 104, 'HERO');

SELECT * FROM movie_cast;



ACT_ID	MOV_ID	ROLE
301	102	HEROINE
301	101	HEROINE
302	102	HERO
303	103	GUEST
304	104	HERO

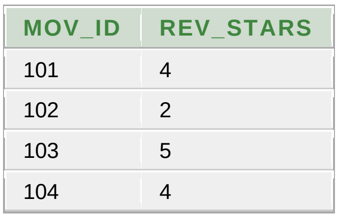
INSERT INTO rating VALUES (101, 4);

INSERT INTO rating VALUES (102, 2);

INSERT INTO rating VALUES (103, 5);

INSERT INTO rating VALUES (104, 4);

```
SELECT * FROM rating;
```



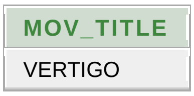
MOV_ID	REV_STARS
101	4
102	2
103	5
104	4

1. 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');
```

(OR)

```
select m.mov_title
from movies m, director d
where m.dir_id = d.dir_id
and d.dir_name = 'HITCHCOCK';
```



MOV_TITLE
VERTIGO

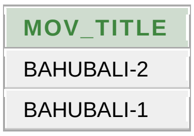
2. Find the movie names where one or more actors acted in two or more movies.

```
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);
```

(OR)

```
select mov_title
from movies
where mov_id in (
            select mov_id
            from movie_cast
            where act_id in (
                        select act_id
```

```
                                        from movie_cast

                                        group by act_id having count (act_id) > 1

                                        )

                  );
```



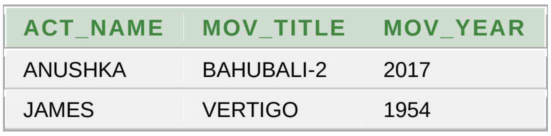
MOV_TITLE
BAHUBALI-2
BAHUBALI-1

3. 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;
```

(OR)

```
select act_name, mov_title, mov_year

from actor

join movie_cast using (act_id)

join movies using (mov_id)

where mov_year not between 2000 and 2015;
```



ACT_NAME	MOV_TITLE	MOV_YEAR
ANUSHKA	BAHUBALI-2	2017
JAMES	VERTIGO	1954

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.

```
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;
```

MOV_TITLE	MAX(REV_STARS)
BAHUBALI-1	2
BAHUBALI-2	4
MEN IN BLACK	4
VERTIGO	5

5. Update rating of all movies directed by ‘Steven Spielberg’ to 5

```
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'));
```

```
1 row(s) updated.
```

```
select * from rating;
```

MOV_ID	REV_STARS
101	4
102	2
103	5
104	5

2. 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.

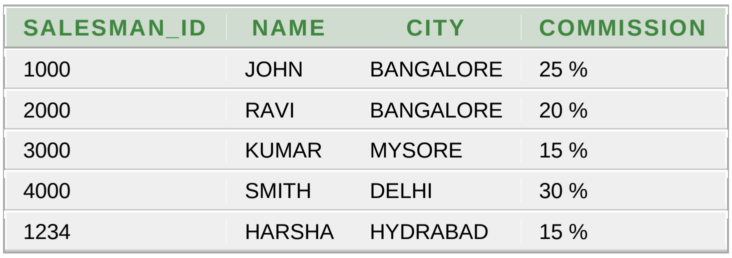
2. Find the name and numbers of all salesmen who had more than one customer.

3. List all the 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.

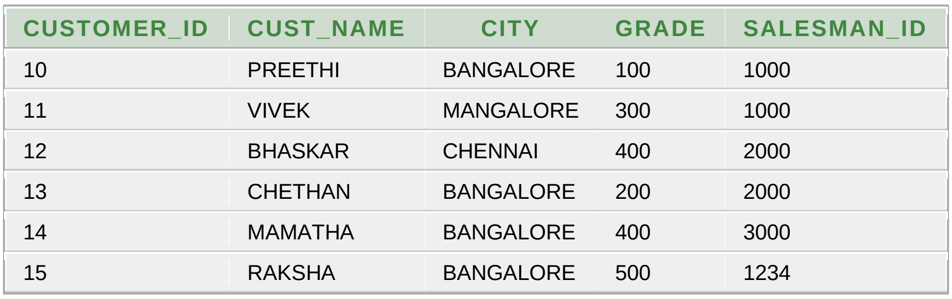
5. Demonstrate the DELETE operation by removing salesman with id 12345. All his orders must also be deleted.

```
create table salesman(
salesman_id number(4) primary key,
name varchar(20),
city varchar(20),
commission varchar(5)
);

create table customer(
customer_id number(4) primary key,
cust_name varchar(20),
city varchar(20),
grade number(5),
salesman_id number(4),
foreign key (salesman_id) references salesman(salesman_id) on delete set null
);

create table orders(
ord_no number(4),
purchase_amt number(6,2),
ord_date date,
customer_id number(4),
salesman_id number(4),
foreign key (salesman_id) references salesman(salesman_id) on delete cascade,
foreign key (customer_id) references customer(customer_id) on delete cascade
);

INSERT INTO salesman VALUES (1000, 'JOHN','BANGALORE','25 %');
INSERT INTO salesman VALUES (2000, 'RAVI','BANGALORE','20 %');
INSERT INTO salesman VALUES (3000, 'KUMAR','MYSORE','15 %');
INSERT INTO salesman VALUES (4000, 'SMITH','DELHI','30 %');
INSERT INTO salesman VALUES (1234, 'HARSHA','HYDRABAD','15 %');
SELECT * FROM salesman;
```



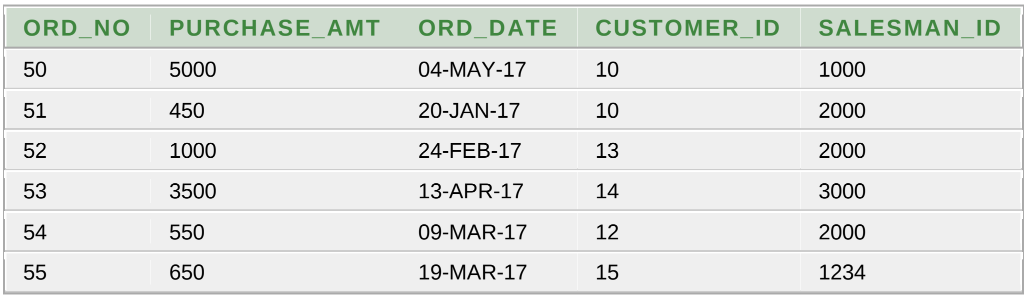
SALESMAN_ID	NAME	CITY	COMMISSION
1000	JOHN	BANGALORE	25 %
2000	RAVI	BANGALORE	20 %
3000	KUMAR	MYSORE	15 %
4000	SMITH	DELHI	30 %
1234	HARSHA	HYDRABAD	15 %

```
INSERT INTO customer VALUES (10, 'PREETHI','BANGALORE', 100, 1000);
INSERT INTO customer VALUES (11, 'VIVEK','MANGALORE', 300, 1000);
INSERT INTO customer VALUES (12, 'BHASKAR','CHENNAI', 400, 2000);
INSERT INTO customer VALUES (13, 'CHETHAN','BANGALORE', 200, 2000);
INSERT INTO customer VALUES (14, 'MAMATHA','BANGALORE', 400, 3000);
INSERT INTO customer VALUES (15, 'RAKSHA','BANGALORE', 500, 1234);
SELECT * FROM customer;
```



CUSTOMER_ID	CUST_NAME	CITY	GRADE	SALESMAN_ID
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
15	RAKSHA	BANGALORE	500	1234

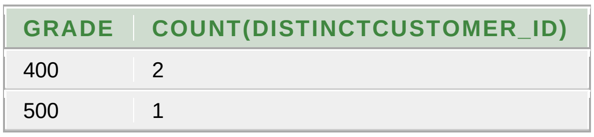
```
INSERT INTO orders VALUES (50, 5000, '04-MAY-17', 10, 1000);
INSERT INTO orders VALUES (51, 450, '20-JAN-17', 10, 2000);
INSERT INTO orders VALUES (52, 1000, '24-FEB-17', 13, 2000);
INSERT INTO orders VALUES (53, 3500, '13-APR-17', 14, 3000);
INSERT INTO orders VALUES (54, 550, '09-MAR-17', 12, 2000);

INSERT INTO orders VALUES (55, 650, '19-MAR-17', 15, 1234);

SELECT * FROM orders;
```



ORD_NO	PURCHASE_AMT	ORD_DATE	CUSTOMER_ID	SALESMAN_ID
50	5000	04-MAY-17	10	1000
51	450	20-JAN-17	10	2000
52	1000	24-FEB-17	13	2000
53	3500	13-APR-17	14	3000
54	550	09-MAR-17	12	2000
55	650	19-MAR-17	15	1234

1. Count the customers with grades above Bangalore’s average.

```
SELECT grade, COUNT (DISTINCT customer_id)

FROM customer

GROUP BY grade

HAVING grade > (SELECT AVG(grade)

FROM customer

WHERE city='BANGALORE');
```



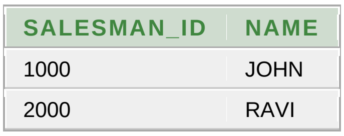
GRADE	COUNT(DISTINCTCUSTOMER_ID)
400	2
500	1

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

```
SELECT salesman_id, name

FROM salesman A

WHERE 1 < (SELECT COUNT (*)

FROM customer

WHERE salesman_id=A.salesman_id);
```

(OR)

```
select salesman_id, name

from salesman

where salesman_id in (select salesman_id from customer group by salesman_id having count (*) > 1);
```



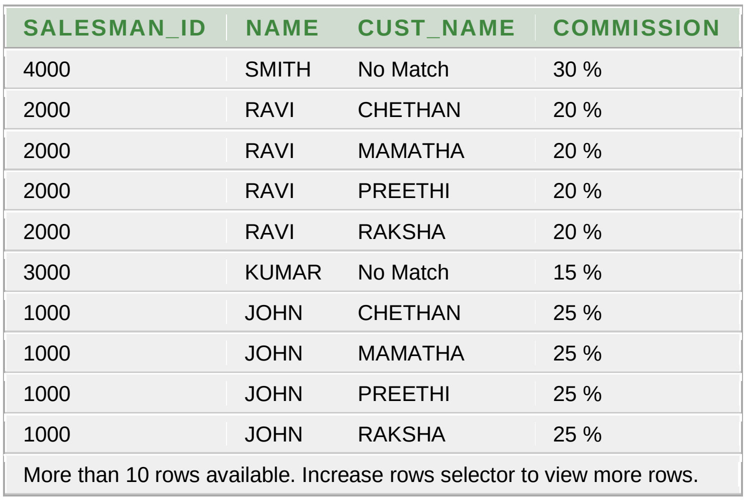
SALESMAN_ID	NAME
1000	JOHN
2000	RAVI

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

```
SELECT SALESMAN.salesman_id, name, cust_name, commission

FROM salesman, customer

WHERE SALESMAN.city = CUSTOMER.city

UNION

SELECT salesman_id, name, 'NO MATCH', commission

FROM salesman

WHERE NOT city = ANY

(SELECT city

FROM customer)

ORDER BY 2 DESC;
```

(OR)

```
select salesman.salesman_id, name, cust_name,commission

from salesman, customer

where salesman.city = customer.city

UNION

select salesman_id, name, 'No Match',commission

from salesman

where city not in (select city from customer)

order by 2 desc;
```



SALESMAN_ID	NAME	CUST_NAME	COMMISSION
4000	SMITH	No Match	30 %
2000	RAVI	CHETHAN	20 %
2000	RAVI	MAMATHA	20 %
2000	RAVI	PREETHI	20 %
2000	RAVI	RAKSHA	20 %
3000	KUMAR	No Match	15 %
1000	JOHN	CHETHAN	25 %
1000	JOHN	MAMATHA	25 %
1000	JOHN	PREETHI	25 %
1000	JOHN	RAKSHA	25 %

More than 10 rows available. Increase rows selector to view more rows.

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

```
CREATE VIEW ELITSALESMAN AS
```

```
SELECT B.ord_date, A.salesman_id, A.name

FROM salesman A, orders B

WHERE A.salesman_id = B.salesman_id

AND B.purchase_amt=(SELECT MAX (purchase_amt)

FROM orders C

WHERE C.ord_date = B.ord_date);
```

```
View created.
```

```
select * from ELITSALESMAN;
```

(OR)

```
create view Highest_order as

select ord_date, salesman_id, name

from orders natural join salesman

where (purchase_amt, ord_date) in (select max(purchase_amt), ord_date from orders

group by ord_date);
```

```
View created.
```

```
select * from Highest_order;
```

ORD_DATE	SALESMAN_ID	NAME
04-MAY-17	1000	JOHN
20-JAN-17	2000	RAVI
09-MAR-17	2000	RAVI
24-FEB-17	2000	RAVI
13-APR-17	3000	KUMAR
19-MAR-17	1234	HARSHA

5. Demonstrate the DELETE operation by removing salesman with id 12345. All his orders must also be deleted.

```
DELETE FROM salesman

WHERE salesman_id=1234;
```

```
1 row(s) deleted.
```

```
SELECT * FROM salesman;
```

SALESMAN_ID	NAME	CITY	COMMISSION
1000	JOHN	BANGALORE	25 %
2000	RAVI	BANGALORE	20 %
3000	KUMAR	MYSORE	15 %
4000	SMITH	DELHI	30 %

3. Consider the schema for College Database:

STUDENT(USN, SName, Address, Phone, Gender)

SEMSEC(SSID, Sem, Sec)

CLASS(USN, SSID)

SUBJECT(Subcode, Title, Sem, Credits)

CIEMARKS(USN, Subcode, SSID, CIE1, CIE2, CIE3, FinalCIE)

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 ‘1DA15CS101’ in all subjects.

4. Calculate the FinalCIE (average of best two test marks) and update the corresponding table for all students.

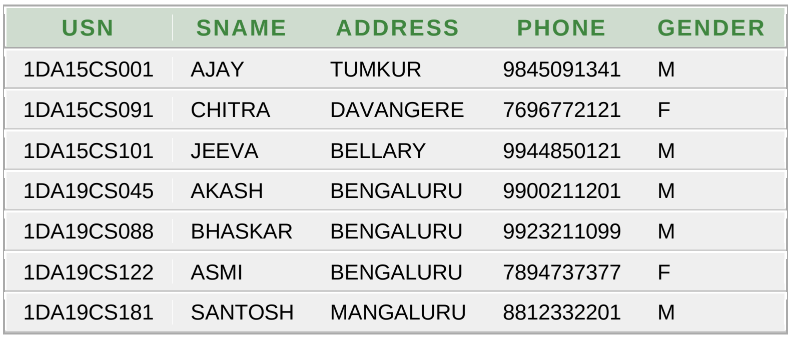
5. Categorize students based on the following criterion:

> **If FinalCIE = 17 to 20 then CAT = ‘Outstanding’**
>
> **If FinalCIE< 12 then CAT = ‘Weak’**
>
> **If FinalCIE = 12 to 16 then CAT = ‘Average’**

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

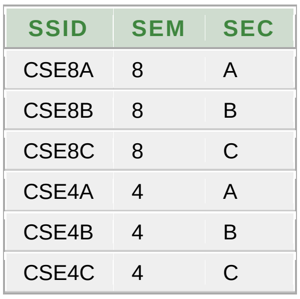
```
create table student(
USN varchar(10),
sname varchar(50),
address varchar(40),
phone number(10),
gender char,
primary key(USN)
);

create table semsec(
SSID varchar(5),
sem number(1),
sec char,
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(6),
title varchar(15),
sem number(1),
credits number(1),
primary key(subcode)
);

create table ciemarks(
USN varchar(10),
subcode varchar(6),
SSID varchar(5),
CIE1 number(2),
CIE2 number(2),
CIE3 number(2),
FinalCIE number(2),
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
('1DA15CS001','AJAY','TUMKUR', 9845091341,'M');
INSERT INTO student VALUES
('1DA15CS091','CHITRA','DAVANGERE',7696772121,'F');
```

```
INSERT INTO student VALUES
('1DA15CS101','JEEVA','BELLARY', 9944850121,'M');
INSERT INTO student VALUES
('1DA19CS045','AKASH','BENGALURU',9900211201,'M');
INSERT INTO student VALUES
('1DA19CS088','BHASKAR','BENGALURU',9923211099,'M');
INSERT INTO student VALUES
('1DA19CS122','ASMI','BENGALURU', 7894737377,'F');
INSERT INTO student VALUES
('1DA19CS181','SANTOSH','MANGALURU',8812332201,'M');
SELECT * FROM student;
```



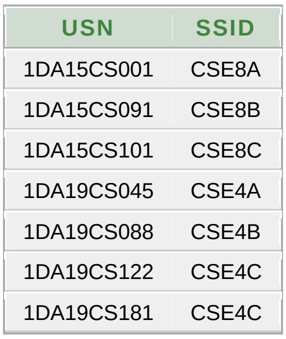
USN	SNAME	ADDRESS	PHONE	GENDER
1DA15CS001	AJAY	TUMKUR	9845091341	M
1DA15CS091	CHITRA	DAVANGERE	7696772121	F
1DA15CS101	JEEVA	BELLARY	9944850121	M
1DA19CS045	AKASH	BENGALURU	9900211201	M
1DA19CS088	BHASKAR	BENGALURU	9923211099	M
1DA19CS122	ASMI	BENGALURU	7894737377	F
1DA19CS181	SANTOSH	MANGALURU	8812332201	M

```
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 ('CSE4A', 4,'A');
INSERT INTO semsec VALUES ('CSE4B', 4,'B');
INSERT INTO semsec VALUES ('CSE4C', 4,'C');
SELECT * FROM semsec;
```



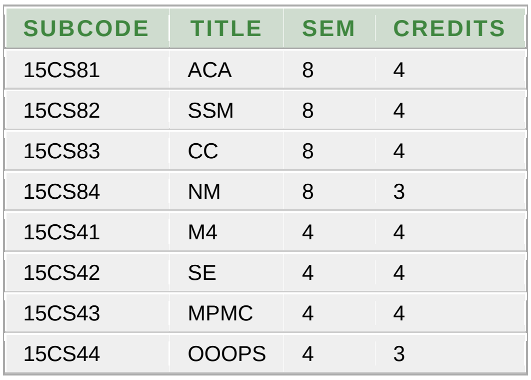
SSID	SEM	SEC
CSE8A	8	A
CSE8B	8	B
CSE8C	8	C
CSE4A	4	A
CSE4B	4	B
CSE4C	4	C

```
INSERT INTO class VALUES ('1DA15CS001','CSE8A');

INSERT INTO class VALUES ('1DA15CS091','CSE8B');

INSERT INTO class VALUES ('1DA15CS101','CSE8C');

INSERT INTO class VALUES ('1DA19CS045','CSE4A');

INSERT INTO class VALUES ('1DA19CS088','CSE4B');

INSERT INTO class VALUES ('1DA19CS122','CSE4C');

INSERT INTO class VALUES ('1DA19CS181','CSE4C');

SELECT * FROM class;
```



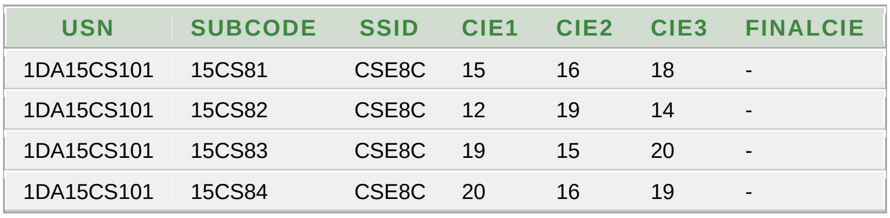
USN	SSID
1DA15CS001	CSE8A
1DA15CS091	CSE8B
1DA15CS101	CSE8C
1DA19CS045	CSE4A
1DA19CS088	CSE4B
1DA19CS122	CSE4C
1DA19CS181	CSE4C

```
INSERT INTO subject VALUES ('15CS81', 'ACA', 8, 4);

INSERT INTO subject VALUES ('15CS82','SSM', 8, 4);

INSERT INTO subject VALUES ('15CS83','CC', 8, 4);

INSERT INTO subject VALUES ('15CS84','NM', 8, 3);

INSERT INTO subject VALUES ('15CS41','M4', 4, 4);

INSERT INTO subject VALUES ('15CS42','SE', 4, 4);

INSERT INTO subject VALUES ('15CS43','MPMC', 4, 4);

INSERT INTO subject VALUES ('15CS44','OOOPS', 4, 3);

SELECT * FROM subject;
```



SUBCODE	TITLE	SEM	CREDITS
15CS81	ACA	8	4
15CS82	SSM	8	4
15CS83	CC	8	4
15CS84	NM	8	3
15CS41	M4	4	4
15CS42	SE	4	4
15CS43	MPMC	4	4
15CS44	OOOPS	4	3

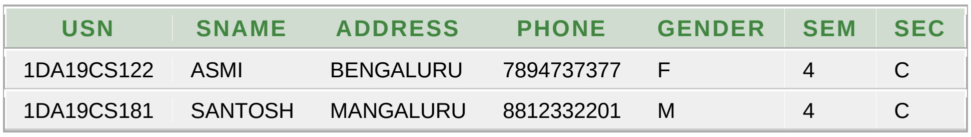
```
INSERT INTO ciemarks (USN, subcode, SSID, CIE1, CIE2, CIE3) VALUES
('1DA15CS101','15CS81','CSE8C', 15, 16, 18);
INSERT INTO ciemarks (USN, subcode, SSID, CIE1, CIE2, CIE3) VALUES
('1DA15CS101','15CS82','CSE8C', 12, 19, 14);
INSERT INTO ciemarks (USN, subcode, SSID, CIE1, CIE2, CIE3) VALUES
('1DA15CS101','15CS83','CSE8C', 19, 15, 20);
INSERT INTO ciemarks (USN, subcode, SSID, CIE1, CIE2, CIE3) VALUES
('1DA15CS101','15CS84','CSE8C', 20, 16, 19);
SELECT * FROM ciemarks;
```



USN	SUBCODE	SSID	CIE1	CIE2	CIE3	FINALCIE
1DA15CS101	15CS81	CSE8C	15	16	18	-
1DA15CS101	15CS82	CSE8C	12	19	14	-
1DA15CS101	15CS83	CSE8C	19	15	20	-
1DA15CS101	15CS84	CSE8C	20	16	19	-

1. 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';
```



USN	SNAME	ADDRESS	PHONE	GENDER	SEM	SEC
1DA19CS122	ASMI	BENGALURU	7894737377	F	4	C
1DA19CS181	SANTOSH	MANGALURU	8812332201	M	4	C

2. 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;
```

SEM	SEC	GENDER	COUNT
4	A	M	1
4	B	M	1
4	C	F	1
4	C	M	1
8	A	M	1
8	B	F	1
8	C	M	1

3. Create a view of Test1 marks of student USN ‘1DA15CS101’ in all subjects.

```
create view stu_cie1_marks as

select CIE1, subcode

from ciemarks

where USN = '1DA15CS101';
```

```
View created.
```

```
Select * from stu_cie1_marks;
```

CIE1	SUBCODE
15	15CS81
12	15CS82
19	15CS83
20	15CS84

4. Calculate the FinalCIE (average of best two test marks) and update the corresponding table for all students.

```
update ciemarks

set FinalCIE = ((CIE1 + CIE2 + CIE3) - LEAST(CIE1, CIE2, CIE3))/2;
```

```
4 row(s) updated.
```

```
select * from ciemarks;
```

USN	SUBCODE	SSID	CIE1	CIE2	CIE3	FINALCIE
1DA15CS101	15CS81	CSE8C	15	16	18	17
1DA15CS101	15CS82	CSE8C	12	19	14	17
1DA15CS101	15CS83	CSE8C	19	15	20	20
1DA15CS101	15CS84	CSE8C	20	16	19	20

5. Categorize students based on the following criterion:

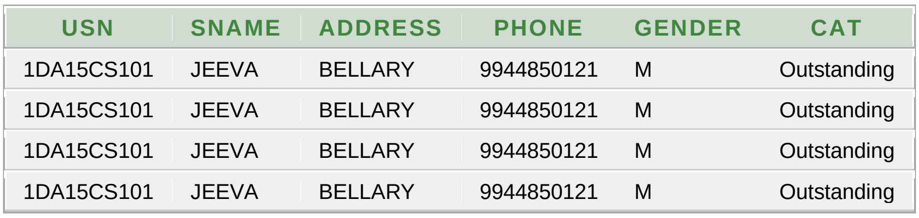
If FinalCIE = 17 to 20 then CAT = ‘Outstanding’

If FinalCIE< 12 then CAT = ‘Weak’

If FinalCIE = 12 to 16 then CAT = ‘Average’

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

```
select s.*,

(case

when m.FinalCIE between 17 and 20 then 'Outstanding'

when m.FinalCIE between 12 and 16 then 'Average'

else 'weak'

end) as CAT

from student s, semsec ss, ciemarks m, subject sub

where s.USN = m.USN and ss.SSID = m.SSID and

sub.subcode = m.subcode and sub.sem = 8;
```



USN	SNAME	ADDRESS	PHONE	GENDER	CAT
1DA15CS101	JEEVA	BELLARY	9944850121	M	Outstanding
1DA15CS101	JEEVA	BELLARY	9944850121	M	Outstanding
1DA15CS101	JEEVA	BELLARY	9944850121	M	Outstanding
1DA15CS101	JEEVA	BELLARY	9944850121	M	Outstanding