NAME: RIYA RAJENDRA JADHAV
ROLL No.: L009

ADVANCE DATABASE MANAGEMENT SYSTEM

Practical 2

Subquery-join operations on Relational Schema

1. Design ERD for the following schema and execute the following Queries on it:

Consider the schema for Movie Database:

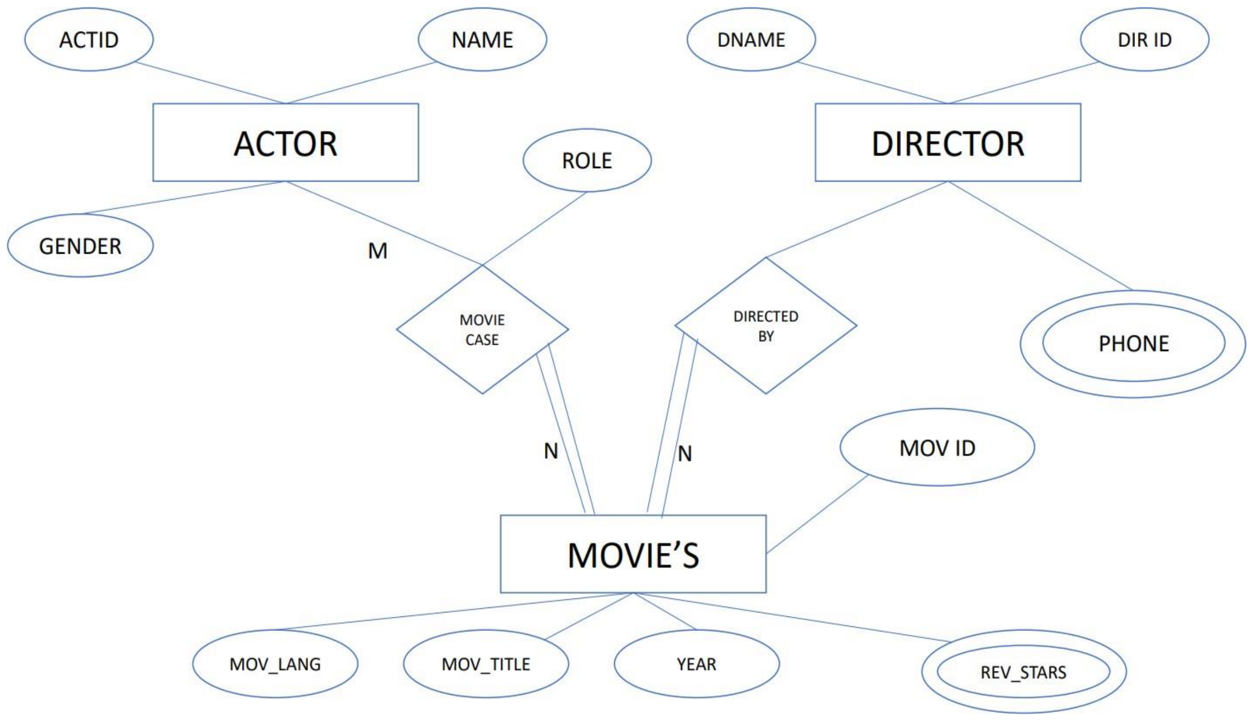
ACTOR (Act_id, Act_Name, Act_Gender)

DIRECTOR (Dir_id, Dir_Name, Dir_Phone)

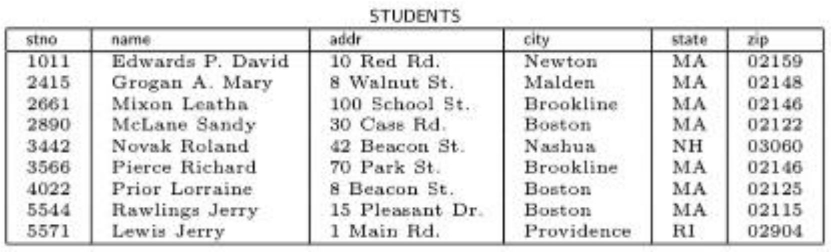
MOVIES (Mov_id, Mov_Title, Mov_Year, Mov_Lang,

Dir_id)MOVIE_CAST (Act_id, Mov_id, Role)

RATING (Mov_id, Rev_Stars)

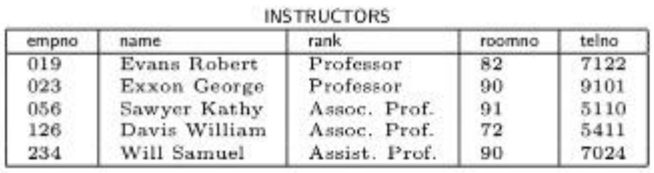


3. Design ERD for the following schema and execute the following Queries on it:



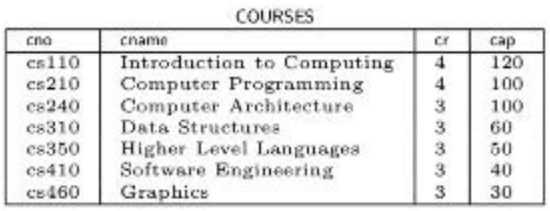
STUDENTS

stno	name	addr	city	state	zip
1011	Edwards P. David	10 Red Rd.	Newton	MA	02159
2415	Grogan A. Mary	8 Walnut St.	Malden	MA	02148
2661	Mixon Leatha	100 School St.	Brookline	MA	02146
2890	McLane Sandy	30 Cass Rd.	Boston	MA	02122
3442	Novak Roland	42 Beacon St.	Nashua	NH	03060
3566	Pierce Richard	70 Park St.	Brookline	MA	02146
4022	Prior Lorraine	8 Beacon St.	Boston	MA	02125
5544	Rawlings Jerry	15 Pleasant Dr.	Boston	MA	02115
5571	Lewis Jerry	1 Main Rd.	Providence	RI	02904



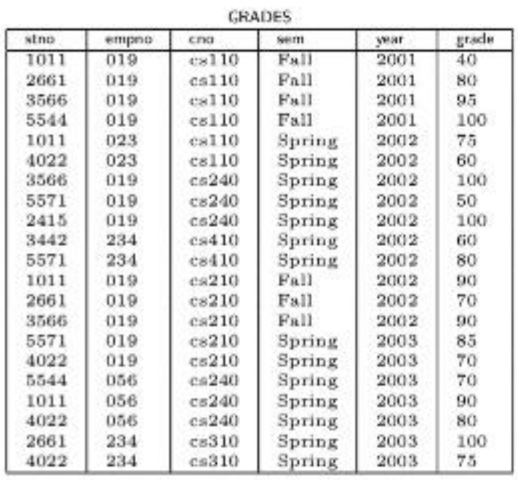
INSTRUCTORS

empno	name	rank	roomno	telno
019	Evans Robert	Professor	82	7122
023	Exxon George	Professor	90	9101
056	Sawyer Kathy	Assoc. Prof.	91	5110
126	Davis William	Assoc. Prof.	72	5411
234	Will Samuel	Assist. Prof.	90	7024



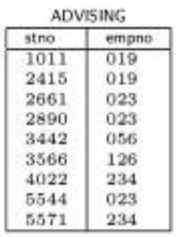
COURSES

cno	cname	cr	cap
cs110	Introduction to Computing	4	120
cs210	Computer Programming	4	100
cs240	Computer Architecture	3	100
cs310	Data Structures	3	60
cs350	Higher Level Languages	3	50
cs410	Software Engineering	3	40
cs460	Graphics	3	30



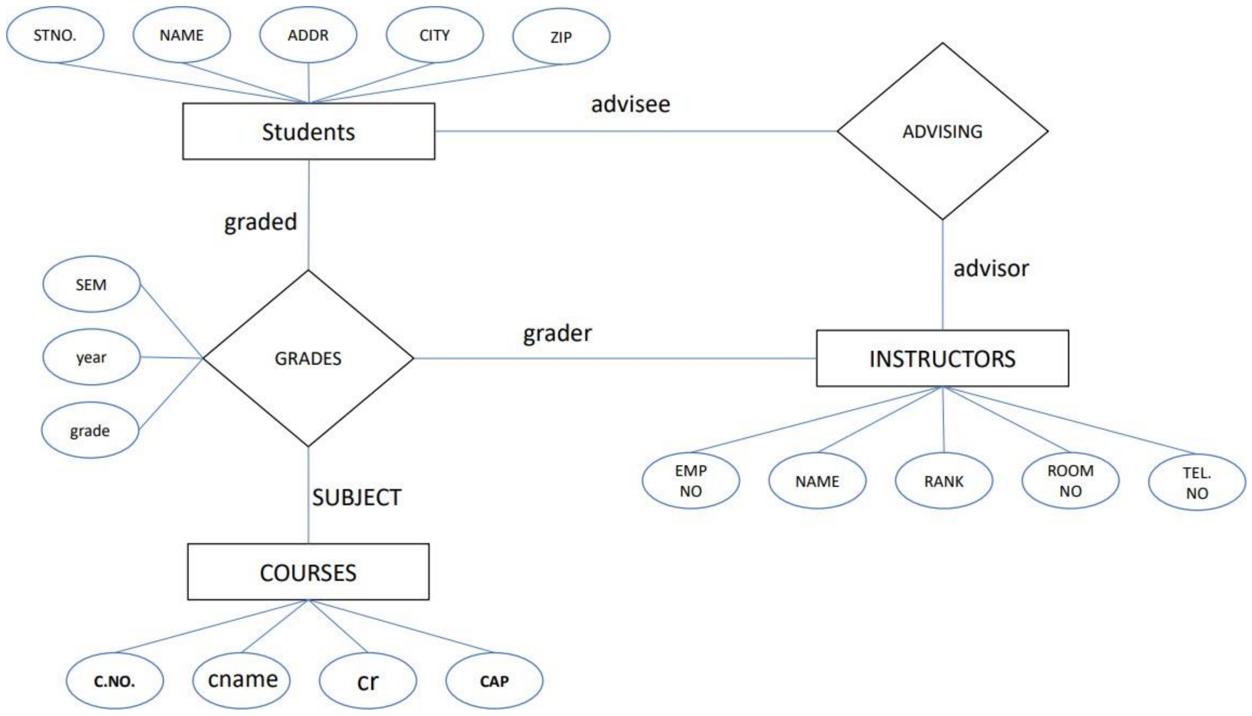
GRADES

stno	empno	cno	sem	year	grade
1011	019	cs110	Fall	2001	40
2661	019	cs110	Fall	2001	80
3566	019	cs110	Fall	2001	95
5544	019	cs110	Fall	2001	100
1011	023	cs110	Spring	2002	75
4022	023	cs110	Spring	2002	60
3566	019	cs240	Spring	2002	100
5571	019	cs240	Spring	2002	50
2415	019	cs240	Spring	2002	100
3442	234	cs410	Spring	2002	60
5571	234	cs410	Spring	2002	80
1011	019	cs210	Fall	2002	90
2661	019	cs210	Fall	2002	70
3566	019	cs210	Fall	2002	90
5571	019	cs210	Spring	2003	85
4022	019	cs210	Spring	2003	70
5544	056	cs240	Spring	2003	70
1011	056	cs240	Spring	2003	90
4022	056	cs240	Spring	2003	80
2661	234	cs310	Spring	2003	100
4022	234	cs310	Spring	2003	75



ADVISING

stno	empno
1011	019
2415	019
2661	023
2890	023
3442	056
3566	126
4022	234
5544	023
5571	234


STNO.
NAME
ADDR
CITY
ZIP
Students
advisee
ADVISING
graded
advisor
SEM
year
grade
GRADES
grader
INSTRUCTORS
EMP NO
NAME
RANK
ROOM NO
TEL. NO
SUBJECT
COURSES
C.NO.
cname
cr
CAP

1. USING (practical 1)

1. Count the customers with grades above NewYork average

```
mysql> SELECT grade, COUNT(*) FROM customer GROUP BY grade HAVING grade >

(SELECTAVG(grade) FROM customer WHERE city = 'New York');

+-------+-----------+

| grade | COUNT(*) |

+-------+-----------+

| 200     |             3 |

| 300     |             2 |

+-------+-----------+

2 rows in set (0.02 sec)
```

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

```
mysql> select salesman_id,name from salesman a where 1<(select count(*) fromcustomer
where salesman_id=a.salesman_id);
  +-------------+-------------+
  | salesman_id   |         name |
  +-------------+-------------+
  | 5001             | James Hoog |
  | 5002             | Nail Knite |
  +-------------+-------------+
  2 rows in set (0.01 sec)
```

3)Demonstrate the DELETE operation by removing salesman with id 1000. All his ordersmust also be deleted

```
mysql> delete from salesman where salesman_id=1000;
Query OK, 0 rows affected (0.00 sec)
```

Q2. Design ERD for the following schema and execute the following Queries on it:

```
mysql> create table Actor(act_id integer primary key,act_name
varchar(100),act_gender varchar(10));
Query OK, 0 rows affected (0.01 sec)

mysql> create table Director(dir_id integer primary key,dir_name
varchar(200),dir_phone varchar(100));
Query OK, 0 rows affected (0.01 sec)

mysql> create table Movies(mov_id integer primary key,mov_title
varchar(255),mov_year year,mov_lang varchar(100),dir_id int, foreign key
(dir_id)references Director(dir_id));
Query OK, 0 rows affected (0.02 sec)

mysql> create table Movie_cast (act_id int,foreign key (act_id) references
Actor(act_id), mov_id int, foreign key(mov_id) references
Movies(mov_id),rolevarchar(100), primary key(act_id,mov_id) );
Query OK, 0 rows affected (0.02 sec)

mysql> create table Rating(mov_id integer primary key , foreign
key(mov_id)references Movies(mov_id),rev_stars integer);
Query OK, 0 rows affected (0.01 sec)

mysql> insert into Actor values(301, 'anuska','f'),
 -> (302,'PRABHAS','M'),
 -> (303,'PUNITH','M'),
 -> (304,'jermy','M');
Query OK, 4 rows affected (0.03 sec)
Records: 4 Duplicates: 0 Warnings: 0

mysql> insert into director values(60, 'rajamouli',8751611001),
 -> (61,'HITCHCOCK', 7766138911),
 -> (62,'FARAN', 9986776531),
 -> (63,'STEVEN SPIELBERG', 8989776530);
Query OK, 4 rows affected (0.00 sec)
```

```
Records: 4 Duplicates: 0 Warnings: 0

mysql> insert into movies values(1001,'BAHUBALI-2', 2017, 'TELAGU', 60),
 -> (1002,'BAHUBALI-2', 2015, 'TELAGU', 60),
 -> (1003,'AKASH', 2008, 'KANNADA', 61),
 -> (1004,'WAR HORSE', 2011, 'ENGLISH', 63);
Query OK, 4 rows affected (0.00 sec)
Records: 4 Duplicates: 0 Warnings: 0

mysql> INSERT INTO MOVIE_CAST VALUES (301, 1002, 'HEROINE'),
 -> (301, 1001, 'HEROINE'),
 -> (303, 1003, 'HERO'),
 -> (303, 1002, 'guest'),
 -> (304, 1004, 'hero');
Query OK, 5 rows affected (0.00 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> INSERT INTO RATING VALUES (1001, 4),
 -> (1002, 2),
 -> (1003, 5),
 -> (1004, 4);
Query OK, 4 rows affected (0.00
sec)Records: 4 Duplicates: 0
Warnings: 0
```

1. List the titles of all movies directed by ‘Hitchcock

```
mysql> select mov_title from movies where dir_id in(select dir_id from director
where dir_name='hitchcock');
+-----------+
| mov_title |
+-----------+
| AKASH     |
+-----------+
1 row in set (0.00 sec)
```

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

```
mysql> 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;
+------------+
| mov_title  |
+------------+
| BAHUBALI-2 |
+------------+
1 row in set (0.00 sec)
```

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

```
mysql> select a.act_name,c.mov_title,c.mov_year from actor a,movie_cast b,movies c
where a.act_id=b.act_id and b.mov_id=c.mov_id and c.mov_year not between 2000 and
2015;
```

```
+----------+-+-----------+----------+
| act_name | mov_title  | mov_year |
+----------+-+-----------+----------+
| anuska   | BAHUBALI-2 | 2017   |
+----------+-+-----------+
-----------+1 row in set (0.00
sec)
```

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

```
mysql> 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_tit
le;
+------------+----------------+
| mov_title  | max(rev_stars) |
+------------+----------------+
| AKASH          |          5        |
| BAHUBALI-   |          4        |
2
| WAR             |          4        |
HORSE
+------------+
--------------+3 rows in set
(0.00 sec)
```

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

```
mysql> 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'));
Query OK, 1 row affected (0.00 sec)
```

```
Rows matched: 1 Changed: 1 Warnings: 0

mysql> select * from rating;

+--------+-----------+

| mov_id | rev_stars |

+--------+-----------+

| 1001   |         4 |

| 1002   |         2 |

| 1003   |         5 |

| 1004   |         5 |

+--------+-----------+

4 rows in set (0.00 sec)
```

Q3. Design ERD for the following schema and execute the following Queries on it:

```
mysql> CREATE TABLE students (
 -> stno INT PRIMARY KEY,
 -> name VARCHAR(50),
 -> addr VARCHAR(255),
 -> city VARCHAR(50),
 -> state VARCHAR(2),
 -> zip VARCHAR(10)
 -> );
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> CREATE TABLE INSTRUCTORS (
 -> empno INT PRIMARY KEY,
 -> name VARCHAR(50),
 -> ranks VARCHAR(20),
 -> roomno VARCHAR(10),
 -> telno VARCHAR(15)
 -> );
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE TABLE COURSES (
 -> cno text PRIMARY KEY,
 -> cname VARCHAR(50),
 -> cr INT,
 -> cap INT
 -> );
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE TABLE GRADES (
 -> stno INT,
 -> empno INT,
 -> cno VARCHAR(50),
 -> sem VARCHAR(10),
 -> year INT,
 -> grade INT,
 -> FOREIGN KEY (stno) REFERENCES students(stno),
 -> FOREIGN KEY (empno) REFERENCES INSTRUCTORS(empno),
 -> FOREIGN KEY (cno) REFERENCES COURSES(cno)
 -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE ADVISING (
 -> stno INT,
 -> empno INT,
 -> PRIMARY KEY (stno, empno),
 -> FOREIGN KEY (stno) REFERENCES students(stno),
 -> FOREIGN KEY (empno) REFERENCES INSTRUCTORS(empno)
 -> );
Query OK, 0 rows affected (0.02 sec)

mysql> insert into students values
 ->(1011,'edwards p. david','10 red rd','newton','MA','02159')
 ->(2415, 'Grogan A. Mary', '8 Walnut St', 'Malden', 'MA', '02148'),
 -> (2661, 'Mixon Leatha', '100 School St', 'Brookline', 'MA', '02146'),
 -> (2890, 'McLane Sandy', '30 Case Rd', 'Boston', 'MA', '02122'),
 -> (3442, 'Novak Roland', '42 Beacon St', 'Nashua', 'NH', '03060'),
 -> (3566, 'Pierce Richard', '70 Park St', 'Brookline', 'MA', '02146'),
 -> (4022, 'Prior Lorraine', '8 Beacon St', 'Boston', 'MA', '02125'),
 -> (5544, 'Rawlings Jerry', '15 Pleasant Dr', 'Boston', 'MA', '02115'),
 -> (5571, 'Lewis Jerry', '1 Main Rd', 'Providence', 'RI', '02904');
```

```
mysql> select * from students;

+------+-------------------+--------------+-+-----------+-------+-----+-+

| stno | name | addr | city | state | zip |

+------+-------------------+--------------+-+-----------+-------+-----+-+

| 1011 | edwards p. david | 10 red rd | newton | MA | 02159 |

| 2415 | Grogan A. Mary | 8 Walnut St | Malden | MA | 02148 |

| 2661 | Mixon Leatha | 100 School St | Brookline | MA | 02146 |

| 2890 | McLane Sandy | 30 Case Rd | Boston | MA | 02122 |

| 3442 | Novak Roland | 42 Beacon St | Nashua | NH | 03060 |

| 3566 | Pierce Richard | 70 Park St | Brookline | MA | 02146 |

| 4022 | Prior Lorraine | 8 Beacon St | Boston | MA | 02125 |

| 5544 | Rawlings Jerry | 15 Pleasant Dr | Boston | MA | 02115 |

| 5571 | Lewis Jerry | 1 Main Rd | Providence | RI | 02904 |

+------+-------------------+--------------+-+-----------+-------+-----+-

+9 rows in set (0.00 sec)


mysql> INSERT INTO instructors VALUES

 -> (19, 'Evans Robert', 'Professor', '82', '7122'),

 -> (23, 'Exxon George', 'Professor', '90', '9101'),

 -> (56, 'Sawyer Kathy', 'Assoc Prof', '91', '5110'),

 -> (126, 'Davis William', 'Assoc Prof', '72', '5411'),

 -> (234, 'Will Samuel', 'Assist Prof', '90',

'7024');Query OK, 5 rows affected (0.01 sec)

Records: 5 Duplicates: 0 Warnings: 0

.
```

```
mysql> select * from instructors;

+------+-+--------------+-------------+---------+------+-+

| empno | name | ranks | roomno | telno |

+------+-+--------------+-------------+---------+------+-+

| 19 | Evans Robert | Professor | 82 | 7122 |

| 23 | Exxon George | Professor | 90 | 9101 |

| 56 | Sawyer Kathy | Assoc Prof | 91 | 5110 |

| 126 | Davis William | Assoc Prof | 72 | 5411 |

| 234 | Will Samuel | Assist Prof | 90 | 7024 |

+------+-+--------------+-------------+---------+------+-

+5 rows in set (0.00 sec)

mysql> insert into courses values

 -> ('cs110', 'Introduction to Computing', 4, 120),

 -> ('cs210', 'Computer Programming', 4, 100),

 -> ('cs240', 'Computer Architecture', 3, 100),

 -> ('cs310', 'Data Structures', 3, 60),

 -> ('cs350', 'Higher Level Languages', 3, 50),

 -> ('cs410', 'Software Engineering', 3, 40),

 -> ('cs460', 'Graphics', 3, 30);

Query  OK,  7  rows  affected  (0.00

sec)    Records:    7    Duplicates:    0

Warnings:  0 mysql>  select  *  from

courses;

+------+-+--------------------------+-----+-+-----+

| cno | cname | cr | cap |

+------+-+--------------------------+-----+-+-----+
| cs110 | Introduction to Computing | 4 | 120 |

| cs210 | Computer Programming | 4 | 100 |

| cs240 | Computer Architecture | 3 | 100 |

| cs310 | Data Structures | 3 | 60 |

| cs350 | Higher Level Languages | 3 | 50 |

| cs410 | Software Engineering | 3 | 40 |

| cs460 | Graphics | 3 | 30 |

+-------+-+---------------------------+-----+-+-

--------+7 rows in set (0.00 sec)

mysql> insert into grades values

 -> (1011, 019, 'cs110', 'Fall', 2001, 40),

 -> (2661, 019, 'cs110', 'Fall', 2001, 80),

 -> (3566, 019, 'cs110', 'Fall', 2001, 95),

 -> (5544, 019, 'cs110', 'Fall', 2001, 100),

 -> (1011, 023, 'cs110', 'Spring', 2002, 75),

 -> (4022, 023, 'cs110', 'Spring', 2002, 60),

 -> (3566, 019, 'cs240', 'Spring', 2002, 100),

 -> (5571, 019, 'cs240', 'Spring', 2002, 50),

 -> (2415, 019, 'cs240', 'Spring', 2002, 100),

 -> (3442, 234,'cs410', 'Spring', 2002, 60),

 -> (5571, 234, 'cs410', 'Spring', 2002, 80),

 -> (1011, 019, 'cs210', 'Fall', 2002, 90),

 -> (2661, 019, 'cs210', 'Fall', 2002, 70),

 -> (3566, 019, 'cs210', 'Fall', 2002, 90),

 -> (5571, 019, 'cs210', 'Spring', 2003, 85),
 -> (4022, 019, 'cs210', 'Spring', 2003, 70),
 -> (5544, 56, 'cs240', 'Spring', 2003, 70),
 -> (1011, 56, 'cs240', 'Spring', 2003, 90),
 -> (4022, 56, 'cs240', 'Spring', 2003, 80),
 -> (2661, 234, 'cs310', 'Spring', 2003, 100),
 -> (4022, 234, 'cs310', 'Spring',2003, 75);
Query OK, 21 rows affected (0.00
sec) Records: 21 Duplicates: 0
Warnings: 0 mysql> select * from
grades;
+______+_______+______-+-______-+-_____+______-+
| stno | empno | cno | sem | year | grade |
+______+_______+______-+-______-+-_____+______-+
| 1011 | 19 | cs110 | Fall | 2001 | 40 |
| 2661 | 19 | cs110 | Fall | 2001 | 80 |
| 3566 | 19 | cs110 | Fall | 2001 | 95 |
| 5544 | 19 | cs110 | Fall | 2001 | 100 |
| 1011 | 23 | cs110 | Spring | 2002 | 75 |
| 4022 | 23 | cs110 | Spring | 2002 | 60 |
| 3566 | 19 | cs240 | Spring | 2002 | 100 |
| 5571 | 19 | cs240 | Spring | 2002 | 50 |
| 2415 | 19 | cs240 | Spring | 2002 | 100 |
| 3442 | 234 | cs410 | Spring | 2002 | 60 |
| 5571 | 234 | cs410 | Spring | 2002 | 80 |
| 1011 | 19 | cs210 | Fall | 2002 | 90 |
| 2661 | 19 | cs210 | Fall | 2002 | 70 |
| 3566 | 19 | cs210 | Fall | 2002 | 90 |

| 5571 | 19 | cs210 | Spring | 2003 | 85 |

| 4022 | 19 | cs210 | Spring | 2003 | 70 |

| 5544 | 56 | cs240 | Spring | 2003 | 70 |

| 1011 | 56 | cs240 | Spring | 2003 | 90 |

| 4022 | 56 | cs240 | Spring | 2003 | 80 |

| 2661 | 234 | cs310 | Spring | 2003 | 100 |

| 4022 | 234 | cs310 | Spring | 2003 | 75 |

+------+-------+------+-+------+-+-----+------+

+21 rows in set (0.00 sec)

mysql> insert into advising values

 -> (1011,019);

 -> (2415,019),

 -> (2661,0023),

 -> (2890,023),

 -> (3442,0056),

 -> (3566,126),

 -> (4022,234),

 -> (5544,023),

 -> (5571,234);

Query OK, 8 rows affected (0.00

sec)Records: 8 Duplicates: 0

Warnings: 0
```

```
mysql> select * from advising;

+------+-------+

| stno | empno |

+------+-------+

| 1011 | 19 |

| 2415 | 19 |

| 2661 | 23 |

| 2890 | 23 |

| 5544 | 23 |

| 3442 | 56 |

| 3566 | 126 |

| 4022 | 234 |

| 5571 | 234 |

+------+-------+

 9 rows in set (0.00 sec)
```

10 . Find the names of students who obtained the highest grade in cs210.

```
mysql> SELECT s.name

 FROM students s

 JOIN grades g ON s.stno = g.stno

 WHERE g.cno = 'cs210' AND g.grade = (SELECT MAX(grade) FROM grades WHERE cno

= 'cs210');

+-----------------+

| name |

+-----------------+

| edwards p. david |

| Pierce Richard |

+-----------------+
```

1. Find the names of students who took only four-credit courses.

```
SELECT s.name

FROM students s

WHERE NOT

EXISTS (

   SELECT *

   FROM grades g

   JOIN courses c ON g.cno =

   c.cnoWHERE g.stno = s.stno

   AND c.cr != 4

);
```

```
+------------------+
| name             |
+------------------+
| Grogan A. Mary  |
| Mixon Leatha     |
| McLane Sandy     |
| Novak Roland    |
| Pierce Richard |
| Prior Lorraine  |
| Rawlings Jerry |
| Lewis Jerry      |
+------------------+
```

2. Find the names of students who took no four-credit courses.

```
mysql>SELECT s.name
from students s
where not exists (
   select *
   from grades g
   join courses c on g.cno = c.cno
   where g.stno = s.stno
   and c.cr = 4
);
```

```
+------------------+
| name             |
+------------------+
| Edwards P. David |
+------------------+
```

3. Find the names of students who took cs210 or cs310.

```
mysql>select distinct s.name from students s
        join grades g on s.stno = g.stno
        where g.cno in ('cs210', 'cs310');
```

```
+----------------+
| name           |
+----------------+
| Edwards P. David |
| Mixon Leatha   |
| Rawlings Jerry |
| Lewis Jerry    |
+----------------+
```

4. Find names of all students who have a cs210 grade higher than the highest grade given incs310 and did not take any course with Prof. Evans.

```
mysql>select s.name
from students s
join grades g1 on s.stno = g1.stno
join courses c1 on g1.cno = c1.cno
left join grades g2 on s.stno = g2.stno
left join courses c2 on g2.cno = c2.cno and c2.cno = 'cs310'
where g1.cno = 'cs210'
```

```
and g1.grade > (
    select max(g.grade)
    from grades g
    join courses c on g.cno = c.cno
    where c.cno = 'cs310'
)
and g2.empno != 19
and g2.empno is null;


+------------------+
| name             |
+------------------+
| Mixon Leatha     |
| Rawlings Jerry   |
| Lewis Jerry      |
+------------------+
```

6. Find the lowest grade of a student who took a course during the spring of 2003.

```
mysql>SELECT MIN(grade) AS
lowest_gradeFROM grades
WHERE sem = 'Spring' AND year = 2003;
+--------------+
| lowest_grade |
+--------------+
| 60           |
+--------------+
```

7. Find the names of students who obtained the highest grade in cs210.

```
mysql>select s.name
```

```
from students s

join grades g on s.stno = g.stno

where g.cno = 'cs210'

and g.grade = (

    select max(grade)

    from grades

    where cno = 'cs210'
);
+-------------------+
| name              |
+-------------------+
| Grogan A. Mary  |
+-------------------+
```

9. Find the highest grade of a student who never took cs110.

```
mysql>SELECT MAX(grade) AS
highest_gradeFROM grades
WHERE stno NOT IN (
   SELECT
   stno FROM
   grades
   WHERE cno = 'cs110'
);
+---------------+
| highest_grade |
+---------------+
| 95            |
+---------------+
```

8. Find the names of students whose advisor did not teach them any course.

```
mysql>SELECT
s.nameFROM
students s
LEFT JOIN advising a ON s.stno = a.stno
LEFT JOIN grades g ON s.stno = g.stno AND a.empno =
g.empnoWHERE g.empno IS NULL;

+--------------------+

| name               |

+--------------------+
| Edwards P. David |
| Mixon Leatha     |
| McLane Sandy     |
| Novak Roland     |
| Pierce Richard   |
| Prior Lorraine   |
| Rawlings Jerry   |
| Lewis Jerry      |
+------------------+
```