MSC DATA SCIENCE & AI
L026
QURESHI AFRIN
PRACTICAL 02 : Subquery-join operations on Relational Schema

```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14
Server version: 5.5.16 MySQL Community Server (GPL)

Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
mysql>#Q1)USING (practical 1)

mysql>#Q1). Count the customers with grades above NewYork’s average;
mysql>select count(customer_name) from customer where grade>(select avg(grade) from
customer where city='new york');
+----------------------+
| count(customer_name) |
+----------------------+
|                    5 |
+----------------------+
1 row in set (0.04 sec)

mysql>#Q2)Find the name and numbers of all salesmen who had more than one customer;
mysql> select salesman_id,name from salesman where salesman_id in (select
salesman_id from customer group by salesman_id having count(salesman_id)>1);
+-------------+------------+
| salesman_id | name       |
+-------------+------------+
|        5001 | James Hoog |
|        5002 | Nail Knite |
+-------------+------------+
2 rows in set (0.00 sec)

mysql>#Q4)Create a view that finds the salesman who has the customer with the
highest order of a day;
mysql> select salesman_id,customer_name from customer where customer_name in
(select customer_name from customer where grade in (select max(grade) from
customer));
+-------------+----------------+
| salesman_id | customer_name  |
+-------------+----------------+
|        5006 | Fabian Johnson |
|        5002 | Julian Green   |
+-------------+----------------+
2 rows in set (0.00 sec)

mysql>#Q5). Demonstrate the DELETE operation by removing salesman with id 1000. All
his orders must also be deleted;

mysql> delete from orders where salesman_id=5001;
Query OK, 3 rows affected (0.00 sec)
mysql> delete from customer where salesman_id=5001;
Query OK, 2 rows affected (0.01 sec)

mysql> select * from customer ;
+-------------+----------------+------------+-------+-------------+
| customer_id | customer_name  | city       | grade | salesman_id |
+-------------+----------------+------------+-------+-------------+
|        3001 | Brad Guzan     | London     |  NULL |        NULL |
|        3003 | Jozy Altidor   | Moscow     |   200 |        5007 |
|        3004 | Fabian Johnson | Paris      |   300 |        5006 |
|        3005 | Graham Zusi    | California |   200 |        5002 |
|        3008 | Julian Green   | London     |   300 |        5002 |
|        3009 | Geoff Cameron  | Berlin     |   100 |        NULL |
+-------------+----------------+------------+-------+-------------+
6 rows in set (0.00 sec)
mysql> use afrin;
Database changed
mysql> show tables;
+-----------------+
| Tables_in_afrin |
+-----------------+
| actor           |
| customer        |
| director        |
| emp             |
| enroll          |
| movie_cast      |
| movies          |
| ord_itemr       |
| orders          |
| rating          |
| salesman        |
| supplier        |
+-----------------+
12 rows in set (0.01 sec)

mysql> select * from actor;
+--------+----------+------------+
| ACT_ID | ACT_NAME | ACT_GENDER |
+--------+----------+------------+
|    301 | ANUSHKA  | F          |
|    302 | PRABHAS  | M          |
|    303 | PUNITH   | M          |
|    304 | JERMY    | M          |
+--------+----------+------------+
4 rows in set (0.00 sec)

mysql> select * from director;
+--------+-----------------+------------+
| DIR_ID | DIR_NAME        | DIR_PHONE  |
+--------+-----------------+------------+
|     60 | RAJAMOULI       | 8751611001 |
|     61 | HITCHCOCK       | 7766138911 |
|     62 | FARAN           | 9986776531 |
|     63 | STEVEN SPIELBERG | 8989776530 |
+--------+-----------------+------------+
4 rows in set (0.00 sec)

mysql> select * from movie_cast;
+--------+--------+---------+
| ACT_ID | MOV_ID | ROLE    |
+--------+--------+---------+
|    301 |   1001 | HEROINE |
|    301 |   1002 | HEROINE |
|    303 |   1002 | GUEST   |
|    303 |   1003 | HERO    |
|    304 |   1004 | HERO    |
+--------+--------+---------+
5 rows in set (0.00 sec)

mysql> select * from movies;
+--------+------------+----------+----------+--------+
| MOV_ID | MOV_TITLE  | MOV_YEAR | MOV_LANG | DIR_ID |
+--------+------------+----------+----------+--------+
|   1001 | BAHUBALI-2 |     2017 | TELAGU   |     60 |
|   1002 | BAHUBALI-1 |     2015 | TELAGU   |     60 |
|   1003 | AKASH      |     2008 | KANNADA  |     61 |
|   1004 | WAR HORSE  |     2011 | ENGLISH  |     63 |
+--------+------------+----------+----------+--------+
4 rows in set (0.00 sec)

mysql> # Q 2);
mysql> # 1 . list the titles of all the movies directed by ' hicthcock';
                                                            mysql>
mysql> select mov_title from movies where dir_id = (select dir_id from director
where dir_name='hitchcock');
+-----------+
| mov_title |
+-----------+
| AKASH     |
+-----------+
1 row in set (0.04 sec)

mysql>
mysql>#Q2) Find the movie names where one or more actors acted in two or more
movies;

mysql> select mov_title from movies where mov_id in (select mov_id frommovie_cast
where act_id in (select act_id from movie_cast group by act_id having
count(act_id)>=2));
+------------+
| mov_title  |
+------------+
| BAHUBALI-2 |
| BAHUBALI-1 |
| AKASH      |
+------------+
3 rows in set (0.04 sec)

mysql>#Q3) 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 inner joinmovie_cast b
on a.act_id = b.act_id inner join movies c on b.mov_id = c.mov_id where c.mov_year
not between 2000 and 2015;
+----------+-----------+----------+
| act_name | mov_title  | mov_year |
+----------+-----------+----------+
| ANUSHKA  | BAHUBALI-2 |     2017 |
+----------+-----------+----------+
1 row in set (0.00 sec)

mysql> select * from rating;
+--------+-----------+
| MOV_ID | REV_STARS |
+--------+-----------+
|   1001 | 4         |
|   1002 | 2         |
|   1003 | 5         |
|   1004 | 4         |
+--------+-----------+
4 rows in set (0.04 sec)

mysql>
mysql> # Q4) Find the title of movies and number of stars for each movie that has
at least onerating and find the highest number of stars that movie received. Sort
the result bymovie title;
mysql>
mysql> select a.mov_title ,max(b.rev_stars) as max_rev_stars from movies a inner
join rating b on a.mov_id=b.mov_id group by a.mov_title havingmax(b.rev_stars)>=1;
+------------+---------------+
| mov_title  | max_rev_stars |
+------------+---------------+
| AKASH      | 5             |
| BAHUBALI-1 | 2             |
| BAHUBALI-2 | 4             |
| WAR HORSE  | 4             |
+------------+---------------+
4 rows in set (0.05 sec)

mysql>
mysql>
mysql>
mysql> select a.mov_title,max(b.rev_stars) as max_revstar from movies ainner join
rating b on a.mov_id=b.mov_id group by a.mov_title having max(b.rev_stars)>=1;
+------------+-------------+
| mov_title  | max_revstar |
+------------+-------------+
| AKASH      | 5           |
| BAHUBALI-1 | 2           |
| BAHUBALI-2 | 4           |
| WAR HORSE  | 4           |
+------------+-------------+
4 rows in set (0.00 sec)

mysql>Q5) Update rating of all movies directed by ‘Steven Spielberg’ to 5;
mysql> select mov_id from movies where dir_id =  (select dir_id from director where
dir_name='Steven Spielberg');
+--------+
| mov_id |
+--------+
|   1004 |
+--------+
1 row in set (0.04 sec)

mysql> update rating set rev_stars=5 where mov_id in (select mov_id from movies
where dir_id =  (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)
```

```
mysql> # Q 3) MAIN Q;
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 (0.05 sec)

mysql> CREATE TABLE COURSES (
    ->     cno TEXT PRIMARY KEY,
    ->     cname VARCHAR(50),
    ->     cr INT,
    ->     cap INT
    -> );
ERROR 1170 (42000): BLOB/TEXT column 'cno' used in key specification without a key
length
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)
    -> );
ERROR 1005 (HY000): Can't create table 'afrin.grades' (errno: 150)
mysql> show tables;
+-----------------+
| Tables_in_afrin |
+-----------------+
| actor           |
| customer        |
| director        |
| emp             |
| enroll          |
| instructors     |
| movie_cast      |
| movies          |
| ord_itemr       |
| orders          |
| rating          |
| salesman        |
| students        |
| supplier        |
+-----------------+
14 rows in set (0.06 sec)

mysql> CREATE TABLE COURSES (
    ->     cno VARCHAR(50) PRIMARY KEY,
    ->     cname VARCHAR(50),
    ->     cr INT,
    ->     cap INT
    -> );
Query OK, 0 rows affected (0.00 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 (0.01 sec)

mysql> show tables;
+-----------------+
| Tables_in_afrin |
+-----------------+
| actor           |
| courses         |
| customer        |
| director        |
| emp             |
| enroll          |
| grades          |
| instructors     |
| movie_cast      |
| movies          |
| ord_itemr       |
| orders          |
| rating          |
| salesman        |
| students        |
| supplier        |
+-----------------+
16 rows in set (0.05 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');
Query OK, 9 rows affected (0.04 sec)
Records: 9  Duplicates: 0  Warnings: 0

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 (0.05 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 (0.04 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, 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);
Query OK, 21 rows affected (0.05 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, 19),
    -> (2415, 19),
    -> (2661, 23),
    -> (2890, 23),
    -> (3442, 56),
    -> (3566, 126),
    -> (4022, 234),
    -> (5544, 23),
    -> (5571, 234);
ERROR 1146 (42S02): Table 'afrin.advising' doesn't exist
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 (0.05 sec)

mysql> INSERT INTO advising VALUES
    -> (1011, 19),
    -> (2415, 19),
    -> (2661, 23),
    -> (2890, 23),
    -> (3442, 56),
    -> (3566, 126),
    -> (4022, 234),
    -> (5544, 23),
    -> (5571, 234);
Query OK, 9 rows affected (0.03 sec)
Records: 9  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)

mysql> # Q 3);
mysql> # for even roll numbers;
mysql> # Q1)1. Find the names of students who took only four-credit courses.
mysql> select a.name from students a inner join grades b on a.stno=b.stno inner
join courses c on b.cno=c.cno where c.cr=4;
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Rawlings Jerry   |
| Edwards P. David |
| Prior Lorraine   |
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Lewis Jerry      |
| Prior Lorraine   |
+------------------+
11 rows in set (0.00 sec)

mysql> # for EVEN roll numbers;
mysql> #Q1;
mysql> select a.name from students a inner join grades b on a.stn=b.stno inner join
courses c on b.cno=c.cno where c.cr=4;
ERROR 1054 (42S22): Unknown column 'a.stn' in 'on clause'
mysql> select a.name from students a inner join grades b on a.stno=b.stno inner
join courses c on b.cno=c.cno where c.cr=4;
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Rawlings Jerry   |
| Edwards P. David |
| Prior Lorraine   |
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Lewis Jerry      |
| Prior Lorraine   |
+------------------+
11 rows in set (0.00 sec)

mysql> #Q2) Find the names of students who took no four-credit courses ;
mysql> select a.names from students a inner join grades b on a.stno=b.stno inner
join courses c on b.cno=c.cno where c.cr!=4;
ERROR 1054 (42S22): Unknown column 'a.names' in 'field list'
mysql> select a.name from students a inner join grades b on a.stno=b.stno inner
join courses c on b.cno=c.cno where c.cr!=4;
+------------------+
| name             |
+------------------+
| Pierce Richard   |
| Lewis Jerry      |
| Grogan A. Mary   |
| Rawlings Jerry   |
| Edwards P. David |
| Prior Lorraine   |
| Mixon Leatha     |
| Prior Lorraine   |
| Novak Roland     |
| Lewis Jerry      |
+------------------+
10 rows in set (0.01 sec)
```

```
mysql> #Q3) Find the names of students who took cs210 or cs310.
mysql> select a.name from students a inner join grades b on a.stno=b.stno inner
join courses c on b.cno=c.cno where c.cno='cs210' or c.cno='cs310';
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Lewis Jerry      |
| Prior Lorraine   |
| Mixon Leatha     |
| Prior Lorraine   |
+------------------+
7 rows in set (0.04 sec)

mysql>
mysql> #Q4)Find names of all students who have a cs210 grade higher than the
highest grade given in cs310 and did not take any course with Prof. Evans;

mysql> select a.name from students a inner join grades b on a.stno=b.stno inner
join courses c on b.cno=c.cno inner join instructors d on b.empno=d.empno where
b.cno='cs210' and b.grade>(select max(grade) from grades where cno='cs310') and
b.empno!=(select empno from instructors where name='Evans Robert' );
Empty set (0.00 sec)

mysql> #Q5) Find course numbers for courses that enrol at least two students; solve
the same query for courses that enroll at least three students;
mysql> select a.cno from courses a inner join grades b on a.cno=b.cno group by
a.cno having count(distinct b.stno)>=2;
+-------+
| cno   |
+-------+
| cs110 |
| cs210 |
| cs240 |
| cs310 |
| cs410 |
+-------+
5 rows in set (0.05 sec)

mysql> select a.cno from courses a inner join grades b on a.cno=b.cno group by
a.cno having count(distinct b.stno)>=3;
+-------+
| cno   |
+-------+
| cs110 |
| cs210 |
| cs240 |
+-------+
3 rows in set (0.01 sec)

mysql>#Q6)  Find the names of students who obtained the highest grade in cs210;
mysql> select a.name from students a inner join grades b on a.stno=b.stno where
b.grade=(SELECT MAX(grade) FROM grades WHERE cno = 'cs210');
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Edwards P. David |
| Pierce Richard   |
+------------------+
3 rows in set (0.01 sec)

mysql> # Q6);
mysql> select a.name from students a inner join grades b on a.stno=b.stno where
b.grade=(SELECT MAX(grade) FROM grades WHERE cno = 'cs210');
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Edwards P. David |
| Pierce Richard   |
+------------------+
3 rows in set (0.04 sec)

mysql> # Q7) Find the names of instructors who teach courses attended by students
      who took a course with an instructor who is an assistant professor;
mysql>  select name from instructors where empno in (select empno from grades where
stno in (select stno from grades where empno=(select empno from instructors where
ranks='Assist Prof')) and empno not in (select empno from instructors where
ranks='Assist Prof'));
+--------------+
| name         |
+--------------+
| Evans Robert |
| Exxon George |
| Sawyer Kathy |
+--------------+
3 rows in set (0.00 sec)

mysql> #Q8)Find the lowest grade of a student who took a course during the spring
of 2003;
mysql>  select min(grade) from grades  where sem='spring' and year=2003;
+------------+
| min(grade) |
+------------+
|         70 |
+------------+
1 row in set (0.01 sec)
```

```
mysql>
mysql> #Q9);
mysql> # Find the names for students such that if prof. Evans teaches a course,
then the student takes that course;
mysql>  select distinct a.name from students a inner join grades b on a.stno=b.stno
inner join instructors c on b.empno=c.empno where c.name='evans robert';
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Grogan A. Mary   |
| Mixon Leatha     |
| Pierce Richard   |
| Prior Lorraine   |
| Rawlings Jerry   |
| Lewis Jerry      |
+------------------+
7 rows in set (0.00 sec)

mysql> #Q10) Find the highest grade of a student who never took cs110.
mysql> select max(grade) from grades where cno!='cs110';
+------------+
| max(grade) |
+------------+
|        100 |
+------------+
1 row in set (0.00 sec)
```