Firdaus Mulla | L007

Mithibai College Department of Computer Science MSc (Data Science and AI)

Practical-2: Subquery-join operations on Relational Schema

Date: 20/12/2024 **Submission Date: 03/12/2024**

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)

Actor Table:

```
mysql> Create table actor(act_id int not null, act_name varchar(50), act_gender varchar(25), PRIMARY KEY(act_id));
Query OK, 0 rows affected (0.24 sec)

mysql> insert into actor values(1, "David", "Male");
Query OK, 1 row affected (0.08 sec)

mysql> insert into actor values(2, "Miller", "Male");
Query OK, 1 row affected (0.05 sec)

mysql> insert into actor values(3, "Marry", "Female");
Query OK, 1 row affected (0.06 sec)

mysql> insert into actor values(4, "Rose", "Female");
Query OK, 1 row affected (0.07 sec)

mysql> insert into actor values(5, "Lulia", "Female");
Query OK, 1 row affected (0.07 sec)

mysql> insert into actor values(6, "Loyd", "Male");
Query OK, 1 row affected (0.05 sec)
```

Director Table:

```
mysql> Create table director(dir_id int not null, dir_name varchar(50), dir_phone varchar(15), PRIMARY KEY(dir_id));
Query OK, 0 rows affected (0.18 sec)

mysql> insert into director values(1000, "Alfred Hitchcock", 1234567890);
Query OK, 1 row affected (0.09 sec)

mysql> insert into director values(2000, "Steven Spielberg", 2345678901);
Query OK, 1 row affected (0.09 sec)

mysql> insert into director values(3000, "Christopher Nolan", 3456789012);
Query OK, 1 row affected (0.10 sec)

mysql> insert into director values(4000, "Martin Scorsese", 4567123480);
Query OK, 1 row affected (0.07 sec)
```

```
mysql> insert into director values(5000, "Charles Darwin", 6789123450);
Query OK, 1 row affected (0.06 sec)

mysql> insert into director values(6000, "John Lucifer", 7890123456);
Query OK, 1 row affected (0.07 sec)
```

Movies Table

```
mysql> Create table movies(mov_id int not null, mov_title varchar(50), mov_year int, mov_lang varchar(50), dir_id int,
   -> PRIMARY KEY(mov_id), FOREIGN KEY(dir_id) REFERENCES director(dir_id));
Query OK, 0 rows affected (0.59 sec)

mysql> Insert into movies(mov_id, mov_title, mov_year, mov_lang, dir_id) values
   -> (1, 'Psycho', 1960, 'English', 1000),
   -> (2, 'Jaws', 1975, 'English', 4000),
   -> (3, 'Inception', 2010, 'Persian', 5000),
   -> (4, 'The Irishman', 2019, 'French', 6000),
   -> (5, 'Titanic', 1997, 'English', 3000),
   -> (6, 'Iron Man', 2002, 'English', 2000);
Query OK, 6 rows affected (0.10 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

Movie Cast Table

```
mysql> Create table movie_cast(act_id int, mov_id int, role varchar(50), PRIMARY KEY(act_id, mov_id), FOREIGN KEY(act_id) REFERENCES actor(act_id), FOREIGN KEY(mov_id) REFERENCES movies(mov_id));
Query OK, 0 rows affected (0.57 sec)

mysql> Insert into movie_cast(act_id, mov_id, role) values
   -> (1, 1, 'Norman Bates'),
   -> (2, 3, 'Mal'),
   -> (3, 3, 'Dom Cobb'),
   -> (4, 5, 'Rose Dewitt Bukater'),
   -> (5, 2, 'Chief Brody'),
   -> (6, 6, 'Charlie');
Query OK, 6 rows affected (0.12 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

Ratings Table:

```
mysql> Create table rating(mov_id int PRIMARY KEY, rev_stars int, FOREIGN KEY(mov_id) REFERENCES movies(mov_id));
Query OK, 0 rows affected (0.48 sec)

mysql> Insert into rating(mov_id, rev_stars) values(1,5),(2,4),(3,5),(4,3),(5,5);
Query OK, 5 rows affected (0.11 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select * from actor;
+--------+----------+------------+
| act_id | act_name | act_gender |
+--------+----------+------------+
|      1 | David    | Male       |
|      2 | Miller   | Male       |
|      3 | Marry    | Female     |
|      4 | Rose     | Female     |
|      5 | Lulia    | Female     |
|      6 | Loyd     | Male       |
+--------+----------+------------+
6 rows in set (0.00 sec)

mysql> select * from director;
+--------+-------------------+------------+
| dir_id | dir_name          | dir_phone  |
+--------+-------------------+------------+
|   1000 | Alfred Hitchcock  | 1234567890 |
|   2000 | Steven Spielberg  | 2345678901 |
|   3000 | Christopher Nolan | 3456789012 |
|   4000 | Martin Scorsese   | 4567123480 |
|   5000 | Charles Darwin    | 6789123450 |
|   6000 | John Lucifer      | 7890123456 |
+--------+-------------------+------------+
6 rows in set (0.00 sec)

mysql> select * from movies;
+--------+--------------+----------+----------+--------+
| mov_id | mov_title    | mov_year | mov_lang | dir_id |
+--------+--------------+----------+----------+--------+
|      1 | Psycho       |     1960 | English  |   1000 |
|      2 | Jaws         |     1975 | English  |   4000 |
|      3 | Inception    |     2010 | Persian  |   5000 |
|      4 | The Irishman |     2019 | French   |   6000 |
|      5 | Titanic      |     1997 | English  |   3000 |
|      6 | Iron Man     |     2002 | English  |   2000 |
+--------+--------------+----------+----------+--------+
6 rows in set (0.00 sec)

mysql> select * from movie_cast;
+--------+--------+---------------------+
| act_id | mov_id | role                |
+--------+--------+---------------------+
|      1 |      1 | Norman Bates        |
|      2 |      3 | Mal                 |
|      3 |      3 | Dom Cobb            |
|      4 |      5 | Rose Dewitt Bukater |
|      5 |      2 | Chief Brody         |
|      6 |      6 | Charlie             |
+--------+--------+---------------------+
6 rows in set (0.00 sec)

mysql> select * from rating;
+--------+-----------+
| mov_id | rev_stars |
+--------+-----------+
|      1 |         5 |
|      2 |         4 |
|      3 |         5 |
|      4 |         3 |
|      5 |         5 |
+--------+-----------+
5 rows in set (0.00 sec)
```

Write SQL queries to

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

```
mysql> Select mov_title from movies Join director on movies.dir_id = director.dir_id where director.dir_name = "Alfred Hitchcock";
+-----------+
| mov_title |
+-----------+
| Psycho    |
+-----------+
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 Distinct m.mov_title from movies m JOIN movie_cast mc on m.mov_id = mc.mov_id
    -> Where mc.act_id In(Select act_id from movie_cast Group By act_id Having Count(Distinct mov_id) >= 2);
Empty set (0.17 sec)
```

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

```
mysql> insert into movie_cast values(3, 6, "Supporting");
mysql> insert into movie_cast values(2, 4, "Lead");
mysql> insert into movie_cast values(2, 5, "Cameo");
Query OK, 1 row affected (0.06 sec)

mysql> Select Distinct a.act_name from actor a
    -> Join movie_cast mc1 On a.act_id = mc1.act_id
    -> Join movies m1 On mc1.mov_id = m1.mov_id
    -> Join movie_cast mc2 On a.act_id = mc2.act_id
    -> Join movies m2 On mc2.mov_id = m2.mov_id
    -> Where m1.mov_year < 2000 And m2.mov_year > 2015;
+----------+
| act_name |
+----------+
| Miller   |
+----------+
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 m.mov_title, Max(r.rev_stars) As highest_stars from movies m
    -> Join rating r On m.mov_id = r.mov_id Group By m.mov_id Order By m.mov_title;
+-------------+---------------+
| mov_title   | highest_stars |
+-------------+---------------+
| Inception   |             5 |
| Jaws        |             4 |
| Psycho       |             5 |
| The Irishman |             3 |
| Titanic      |             5 |
+--------------+---------------+
5 rows in set (0.00 sec)
```

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

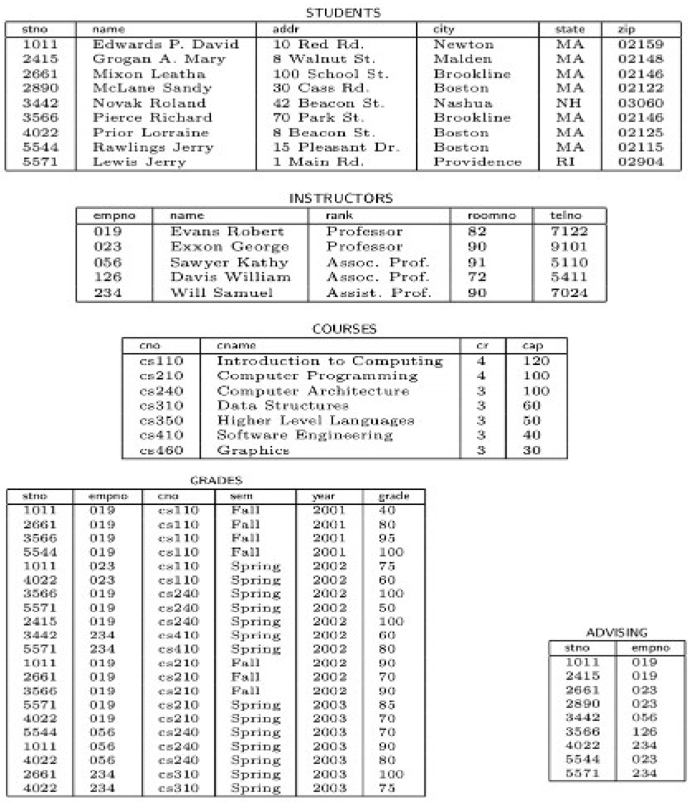
```
mysql> select * from rating;
+--------+-----------+
| mov_id | rev_stars |
+--------+-----------+
|      1 |         5 |
|      2 |         4 |
|      3 |         5 |
|      4 |         3 |
|      5 |         5 |
|      6 |         3 |
+--------+-----------+
6 rows in set (0.00 sec)
```

```
mysql> Update rating r Join movies m on r.mov_id = m.mov_id
    -> Join director d On m.dir_id = d.dir_id
    -> Set r.rev_stars = 5 Where d.dir_name = "Steven Spielberg";
Query OK, 1 row affected (0.08 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> select * from rating;
+--------+-----------+
| mov_id | rev_stars |
+--------+-----------+
|      1 |         5 |
|      2 |         4 |
|      3 |         5 |
|      4 |         3 |
|      5 |         5 |
|      6 |         5 |
+--------+-----------+
6 rows in set (0.00 sec)
```

2. <u>Design ERD for the following schema and execute the following Queries on it:</u>

Firdaus Mulla | L007



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

Students Table:

```
mysql> Create table students(stno int not null PRIMARY KEY, std_name varchar(25),
std_addr varchar(50), city varchar(25), state varchar(15), zip int);
Query OK, 0 rows affected (0.63 sec)

mysql> insert into students(stno, std_name, std_addr, city, state, zip) 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 (0.11 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> select * from students;
+------+------------------+-----------------+------------+-------+------+
| stno | std_name         | std_addr        | city       | state | zip  |
+------+------------------+-----------------+------------+-------+------+
| 1011 | Edwards P. David | 10 Red Rd       | Newton     | MA    | 2159 |
| 2415 | Grogan A. Mary   | 8 Walnut St.    | Malden     | MA    | 2148 |
| 2661 | Mixon Leatha     | 100 School St.  | Brookline  | MA    | 2146 |
| 2890 | McLane Sandy     | 30 Case Rd.     | Boston     | MA    | 2122 |
| 3442 | Novak Roland     | 42 Beacon St.   | Nashua     | NH    | 3060 |
| 3566 | Pierce Richard   | 70 Park St.     | Brookline  | MA    | 2146 |
| 4022 | Prior Lorraine   | 8 Beacon St..   | Boston     | MA    | 2125 |
| 5544 | Rawlings Jerry   | 15 Pleasant Dr. | Boston     | MA    | 2115 |
| 5571 | Lewis Jerry      | 1 Main Rd       | Providence | RI    | 2904 |
+------+------------------+-----------------+------------+-------+------+
9 rows in set (0.00 sec)
```

Instructors Table:

```
mysql> Create table instructors(empno int not null PRIMARY KEY, inst_name varchar(25), inst_rank varchar(25), roomno int, tellno int);
Query OK, 0 rows affected (0.32 sec)

mysql> Insert into instructors(empno, inst_name, inst_rank, roomno, tellno) values
    -> (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", "Assit. Prof.", 90, 7024);
Query OK, 5 rows affected (0.10 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select * from instructors;
+-------+---------------+--------------+--------+--------+
| empno | inst_name     | inst_rank    | roomno | tellno |
+-------+---------------+--------------+--------+--------+
|    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   | Assit. Prof. |     90 |   7024 |
+-------+---------------+--------------+--------+--------+
5 rows in set (0.00 sec)
```

Courses Table:

```
mysql> Create table courses(cno varchar(15) not null PRIMARY KEY, cname varchar(25), cr int, cap int);
Query OK, 0 rows affected (0.28 sec)

mysql> insert into courses(cno, cname, cr, cap) 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 (0.10 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)
```

Grades Table:

```
mysql> Create table grades(stno int, empno int, cno varchar(25), sem varchar(25), 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 (0.26 sec)

mysql> insert into grades(stno, empno, cno, sem, year, grade) 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, 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);
Query OK, 21 rows affected (0.14 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)
```

Advising Table:

```
mysql> Create table advising(stno int, empno int, FOREIGN KEY(stno) References students(stno), FOREIGN KEY(empno) References instructors(empno));
Query OK, 0 rows affected (0.26 sec)

mysql> insert into advising(stno, empno) values
    -> (1011, 019),
    -> (2415, 019),
    -> (2661, 023),
    -> (2890, 023),
    -> (3442, 056),
    -> (3566, 126),
    -> (4022, 234),
    -> (5544, 023),
    -> (5571, 234);
Query OK, 9 rows affected (0.09 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> select * from advising;
+------+-------+
| stno | empno |
+------+-------+
| 1011 |    19 |
| 2415 |    19 |
| 2661 |    23 |
| 2890 |    23 |
| 3442 |    56 |
| 3566 |   126 |
| 4022 |   234 |
| 5544 |    23 |
| 5571 |   234 |
+------+-------+
9 rows in set (0.00 sec)
```

For odd rollnumbers(any 10)

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

```
mysql> Select Distinct s.std_name from students s
    -> Join grades g On s.stno = g.stno Join courses c On g.cno = c.cno
    -> Where c.cr = 4;
+------------------+
| std_name         |
+------------------+
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Rawlings Jerry   |
| Prior Lorraine   |
| Lewis Jerry      |
+------------------+
6 rows in set (0.00 sec)
```

2. Find the names of students who took every four-credit course.

```
Select Count(*) from courses Where cr = 4)' at line 3
mysql> Select s.std_name from students s
    -> Join grades g On s.stno = g.stno Join courses c On g.cno = c.cno
    -> Where c.cr = 4 Group By s.stno Having Count(Distinct c.cno) = (
    -> Select Count(*) from courses Where cr = 4);
+------------------+
| std_name         |
+------------------+
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Prior Lorraine   |
+------------------+
4 rows in set (0.07 sec)
```

3. Find the names of students who took a course with an instructor who is also their advisor.

```
mysql> Select Distinct s.std_name from students s
    -> Join grades g On s.stno = g.stno
    -> Join instructors i On g.empno = i.empno
    -> Join advising a On s.stno = a.stno
    -> Where a.empno = g.empno;
+------------------+
| std_name         |
+------------------+
| Edwards P. David |
| Grogan A. Mary   |
| Prior Lorraine   |
| Lewis Jerry      |
+------------------+
4 rows in set (0.00 sec)
```

4. Find the names of students who took cs210 and cs310.
mysql> Select s.std_name from students s
-> Join grades g On s.stno = g.stno
-> Where g.cno In ("cs210", "cs310") Group By s.stno Having Count(Distinct g.cno) = 2;

```
+----------------+
| std_name       |
+----------------+
| Mixon Leatha   |
| Prior Lorraine |
+----------------+
2 rows in set (0.04 sec)
```

5. Find the names of all students whose advisor is not a full professor.
mysql> Select s.std_name from students s
-> Join advising a On s.stno = a.stno
-> Join instructors i On a.empno = i.empno
-> Where i.inst_rank != "Professor";

```
+----------------+
| std_name       |
+----------------+
| Novak Roland   |
| Pierce Richard |
| Prior Lorraine |
| Lewis Jerry    |
+----------------+
4 rows in set (0.00 sec)
```

6. Find instructors who taught students who are advised by another instructor who shares the same room.
mysql> Select Distinct i.inst_name
-> From instructors i
-> Join grades g On g.empno = i.empno
-> Join advising a On a.stno = g.stno
-> Join instructors i2 ON a.empno = i2.empno
-> Where i.empno != a.empno
->   AND i.roomno = i2.roomno;

```
+--------------+
| inst_name    |
+--------------+
| Exxon George |
| Will Samuel  |
+--------------+
2 rows in set (0.06 sec)
```

7. Find course numbers for courses that enroll exactly two students;
mysql> Select c.cno from courses c
-> Join grades g On g.cno = c.cno
-> Group By c.cno Having Count(Distinct g.stno) = 2;

```
+-------+
| cno   |
+-------+
| cs310 |
| cs410 |
+-------+
2 rows in set (0.04 sec)
```

8. Find the names of all students for whom no other student lives in the same city.
mysql> Select std_name from students Where city Not In(Select city from students Group By city Having Count(stno) > 1);

```
+------------------+
| std_name         |
+------------------+
| Edwards P. David |
| Grogan A. Mary   |
| Novak Roland     |
| Lewis Jerry      |
+------------------+
4 rows in set (0.03 sec)
```

9. Find course numbers of courses taken by students who live in Boston and which are taught by an associate professor.
mysql> Select Distinct c.cno from students s
-> Join grades g On s.stno = g.stno
-> Join courses c On g.cno = c.cno
-> Join instructors i On g.empno = i.empno
-> Where s.city = "Boston" and i.inst_rank = "Assoc. Prof.";

```
+-------+
| cno   |
+-------+
| cs240 |
+-------+
1 row in set (0.00 sec)
```

10. Find the telephone numbers of instructors who teach a course taken by any student who lives in Boston.
mysql> Select Distinct i.tellno from instructors i
-> Join grades g On i.empno = g.empno
-> Join students s On g.stno = s.stno
-> Where s.city = "Boston";

```
+--------+
| tellno |
+--------+
|  9101 |
|  7122 |
|  5110 |
|  7024 |
+--------+
4 rows in set (0.00 sec)
```

11. Find names of students who took every course taken by Richard Pierce.

```
mysql> SELECT s.std_name
    -> FROM students s
    -> JOIN grades g ON s.stno = g.stno
    -> WHERE g.cno IN (
    ->     SELECT g.cno
    ->     FROM grades g
    ->     JOIN students s ON g.stno = s.stno
    ->     WHERE s.std_name = 'Pierce Richard'
    -> )
    -> GROUP BY s.stno
    -> HAVING COUNT(DISTINCT g.cno) = (
    ->     SELECT COUNT(DISTINCT g.cno)
    ->     FROM grades g
    ->     JOIN students s ON g.stno = s.stno
    ->     WHERE s.std_name = 'Pierce Richard'
    -> );
+------------------+
| std_name         |
+------------------+
| Edwards P. David |
| Pierce Richard   |
| Prior Lorraine   |
+------------------+
3 rows in set (0.01 sec)
```

12. Find the names of students who took only one course.

```
mysql> Select s.std_name from students s
    -> Join grades g On s.stno = g.stno
    -> Group By s.stno Having Count(Distinct g.cno) = 1;
+----------------+
| std_name       |
+----------------+
| Grogan A. Mary |
| Novak Roland   |
+----------------+
2 rows in set (0.00 sec)
```

13. Find the names of instructors who teach no course.

```
mysql> Select i.inst_name from instructors i
    -> Left Join grades g On i.empno = g.empno /* Left Join ensures that all instructors are includes even if they do not have any records in the grade table */
    -> Group By i.empno Having Count(g.cno) = 0;
+---------------+
| inst_name     |
+---------------+
| Davis William |
+---------------+
1 row in set (0.00 sec)
```

14. Find the names of the instructors who taught only one course during the spring semester of 2001.

```
mysql> Select i.inst_name from instructors i
    -> Join grades g On i.empno = g.empno
    -> Where g.sem = "Spring" And g.year = 2001 Group By i.empno Having Count(Distinct g.cno) = 1;
Empty set (0.00 sec)
```

****