MSC-DSAI : DBMS

Pradnya Tagad

L037

Practical-2: Subquery-join operations on Relational Schema

1. USING (practical 1)

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

```
mysql> SELECT COUNT(*)
    -> FROM customer
    -> WHERE grade > (SELECT AVG(grade) FROM customer WHERE city = 'Bangalore');
+----------+
| COUNT(*) |
+----------+
|        0 |
+----------+
1 row in set (0.04 sec)
```

2. 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(customer_id) > 1
    -> );
+-------------+------------+
| salesman_id | name       |
+-------------+------------+
|        5003 | Lauson Hen |
+-------------+------------+
1 row in set (0.04 sec)
```

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

```
mysql>
mysql> SELECT salesman_id, name, city, 'Has Customer' AS status
    -> FROM salesman
    -> WHERE city IN (SELECT DISTINCT city FROM customer)
    -> UNION
    -> SELECT salesman_id, name, city, 'No Customer' AS status
    -> FROM salesman
    -> WHERE city NOT IN (SELECT DISTINCT city FROM customer);
+-------------+------------+----------+--------------+
| salesman_id | name       | city     | status       |
+-------------+------------+----------+--------------+
|        5001 | James Hoog | New York | Has Customer |
|        5002 | Nail knite | Paris    | Has Customer |
|        5005 | Pit Alex   | London   | Has Customer |
|        5006 | MC Lyon    | Paris    | Has Customer |
|        5003 | Lauson Hen |          | No Customer  |
|        5007 | Paul Adam  | Rome     | No Customer  |
+-------------+------------+----------+--------------+
6 rows in set (0.00 sec)
```

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

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

```
mysql> DELETE FROM orders
    -> WHERE customer_id IN (SELECT customer_id FROM customers WHERE salesman_id = 1000);
ERROR 1146 (42S02): Table 'pradnya.customers' doesn't exist
mysql> DELETE FROM orders
    -> WHERE customer_id IN (SELECT customer_id FROM customer WHERE salesman_id = 1000);
Query OK, 0 rows affected (0.04 sec)

mysql> desc orders;
+-------------+---------------+------+-----+---------+----------------+
| Field       | Type          | Null | Key | Default | Extra          |
+-------------+---------------+------+-----+---------+----------------+
| order_no    | int           | NO   | PRI | NULL    | auto_increment |
| purch_amt   | decimal(10,2) | NO   |     | NULL    |                |
| order_date  | date          | NO   |     | NULL    |                |
| customer_id | int           | YES  | MUL | NULL    |                |
| salesman_id | int           | YES  | MUL | NULL    |                |
+-------------+---------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

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)

```
mysql> CREATE TABLE ACTOR (
    ->     Act_id INT PRIMARY KEY,
    ->     Act_Name VARCHAR(100),
    ->     Act_Gender VARCHAR(10)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> desc ACTOR;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| Act_id     | int          | NO   | PRI | NULL    |       |
| Act_Name   | varchar(100) | YES  |     | NULL    |       |
| Act_Gender | varchar(10)  | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> INSERT INTO ACTOR (Act_id, Act_Name, Act_Gender) VALUES
    -> (1, 'Tiger shroff', 'Male'),
    -> (2, 'Shradha Kapoor ', 'Female'),
    -> (3, 'Sanjay Datt', 'Male'),
    -> (4, 'Katrina Kafe', 'Female');
Query OK, 4 rows affected (0.01 sec)
```

```
mysql> select * from ACTOR;
+--------+----------------+------------+
| Act_id | Act_Name       | Act_Gender |
+--------+----------------+------------+
|      1 | Tiger shroff   | Male       |
|      2 | Shradha Kapoor | Female     |
|      3 | Sanjay Datt    | Male       |
|      4 | Katrina Kafe   | Female     |
+--------+----------------+------------+
4 rows in set (0.00 sec)
```

DIRECTOR (Dir_id, Dir_Name, Dir_Phone)

```
mysql> CREATE TABLE DIRECTOR (
    ->     Dir_id INT PRIMARY KEY,
    ->     Dir_Name VARCHAR(100),
    ->     Dir_Phone VARCHAR(15)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc DIRECTOR;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| Dir_id    | int          | NO   | PRI | NULL    |       |
| Dir_Name  | varchar(100) | YES  |     | NULL    |       |
| Dir_Phone | varchar(15)  | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> INSERT INTO DIRECTOR (Dir_id, Dir_Name, Dir_Phone) VALUES
    -> (1, 'Raj Kapoor', '9876546780'),
    -> (2, 'Satyajit Ray', '9870987680'),
    -> (3, 'Guru Datt', '9098765678');
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> select * from DIRECTOR;
+--------+-------------+------------+
| Dir_id | Dir_Name    | Dir_Phone  |
+--------+-------------+------------+
|      1 | Raj Kapoor  | 9876546780 |
|      2 | Satyajit Ray | 9870987680 |
|      3 | Guru Datt   | 9098765678 |
+--------+-------------+------------+
3 rows in set (0.00 sec)
```

MOVIES (Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id)

```
mysql> CREATE TABLE MOVIES (
    ->     Mov_id INT PRIMARY KEY,
    ->     Mov_Title VARCHAR(255),
    ->     Mov_Year INT,
    ->     Mov_Lang VARCHAR(50),
    ->     Dir_id INT,
    ->     FOREIGN KEY (Dir_id) REFERENCES DIRECTOR(Dir_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> desc MOVIES;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| Mov_id    | int          | NO   | PRI | NULL    |       |
| Mov_Title | varchar(255) | YES  |     | NULL    |       |
| Mov_Year  | int          | YES  |     | NULL    |       |
| Mov_Lang  | varchar(50)  | YES  |     | NULL    |       |
| Dir_id    | int          | YES  | MUL | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

```
mysql> INSERT INTO MOVIES (Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id) VALUES
    -> (1, 'Titanic', 1997, 'English', 1),
    -> (2, 'Inception', 2010, 'English', 2),
    -> (3, 'Interstellar', 2014, 'English', 2),
    -> (4, 'Jurassic Park', 1993, 'English', 3);
Query OK, 4 rows affected (0.04 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql>
mysql> select * from MOVIES;
+--------+--------------+----------+----------+--------+
| Mov_id | Mov_Title    | Mov_Year | Mov_Lang | Dir_id |
+--------+--------------+----------+----------+--------+
|      1 | Titanic      |     1997 | English  |      1 |
|      2 | Inception    |     2010 | English  |      2 |
|      3 | Interstellar |     2014 | English  |      2 |
|      4 | Jurassic Park |     1993 | English  |      3 |
+--------+--------------+----------+----------+--------+
4 rows in set (0.00 sec)
```

MOVIE_CAST (Act_id, Mov_id, Role)

```
mysql> CREATE TABLE MOVIE_CAST (
    ->     Act_id INT,
    ->     Mov_id INT,
    ->     Role VARCHAR(100),
    ->     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 (0.10 sec)

mysql> desc MOVIE_CAST;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| Act_id | int         | NO   | PRI | NULL    |       |
| Mov_id | int         | NO   | PRI | NULL    |       |
| Role   | varchar(100)| YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

```
mysql> INSERT INTO MOVIE_CAST (Act_id, Mov_id, Role) VALUES
    -> (1, 1, 'Jack Dawson'),
    -> (2, 1, 'Rose DeWitt Bukater'),
    -> (3, 2, 'Dom Cobb'),
    -> (2, 2, 'Mal Cobb'),
    -> (4, 4, 'Dr. Alan Grant'),
    -> (1, 3, 'Dr. Mann');
Query OK, 6 rows affected (0.04 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select * from MOVIE_CAST;
+--------+--------+--------------------+
| Act_id | Mov_id | Role               |
+--------+--------+--------------------+
|      1 |      1 | Jack Dawson        |
|      1 |      3 | Dr. Mann           |
|      2 |      1 | Rose DeWitt Bukater |
|      2 |      2 | Mal Cobb           |
|      3 |      2 | Dom Cobb           |
|      4 |      4 | Dr. Alan Grant     |
+--------+--------+--------------------+
6 rows in set (0.00 sec)
```

RATING (Mov_id, Rev_Stars)

```
mysql> CREATE TABLE RATING (
    ->     Mov_id INT,
    ->     Rev_Stars INT,
    ->     PRIMARY KEY (Mov_id),
    ->     FOREIGN KEY (Mov_id) REFERENCES MOVIES(Mov_id)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> desc RATING;
+-----------+------+------+-----+---------+-------+
| Field     | Type | Null | Key | Default | Extra |
+-----------+------+------+-----+---------+-------+
| Mov_id    | int  | NO   | PRI | NULL    |       |
| Rev_Stars | int  | YES  |     | NULL    |       |
+-----------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> INSERT INTO RATING (Mov_id, Rev_Stars) VALUES
    -> (1, 5),
    -> (2, 4),
    -> (3, 5),
    -> (4, 4);
Query OK, 4 rows affected (0.05 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from RATING;
+--------+-----------+
| Mov_id | Rev_Stars |
+--------+-----------+
|      1 |         5 |
|      2 |         4 |
|      3 |         5 |
|      4 |         4 |
+--------+-----------+
4 rows in set (0.00 sec)
```

Write SQL queries to:

1.List the titles of all movies directed by ‘Raj Kapoor’.

```
mysql> SELECT Mov_Title
    -> FROM MOVIES
    -> WHERE Dir_id = (SELECT Dir_id FROM DIRECTOR WHERE Dir_Name = 'Raj Kapoor');
+-----------+
| Mov_Title |
+-----------+
| Titanic   |
+-----------+
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(*) > 1
    -> );
+--------------+
| Mov_Title    |
+--------------+
| Titanic      |
| Inception    |
| Interstellar |
+--------------+
3 rows 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>
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        |
+-----------------+
| Tiger shroff    |
| Shradha Kapoor  |
+-----------------+
2 rows 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, m.Mov_Title
    -> ORDER BY m.Mov_Title;
+---------------+---------------+
| Mov_Title     | Highest_Stars |
+---------------+---------------+
| Inception     |             4 |
| Interstellar  |             5 |
| Jurassic Park |             4 |
| Titanic       |             5 |
+---------------+---------------+
4 rows in set (0.00 sec)
```

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

```
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, 0 rows affected (0.00 sec)
Rows matched: 0  Changed: 0  Warnings: 0
```

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

For odd rollnumbers(any 10):

- **Students:**

```
mysql> CREATE TABLE Students (
    ->     st_no INT PRIMARY KEY,
    ->     name VARCHAR(50),
    ->     addr VARCHAR(100),
    ->     city VARCHAR(50),
    ->     state VARCHAR(20),
    ->     zip_no VARCHAR(15)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> desc Students;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| st_no  | int         | NO   | PRI | NULL    |       |
| name   | varchar(50) | YES  |     | NULL    |       |
| addr   | varchar(100)| YES  |     | NULL    |       |
| city   | varchar(50) | YES  |     | NULL    |       |
| state  | varchar(20) | YES  |     | NULL    |       |
| zip_no | varchar(15) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Students (st_no, name, addr, city, state, zip_no) VALUES
    -> (1011, 'Edwards P. David', '10 Red Rd.', 'Newton', 'MA', 02159),
    -> (2415, 'Grigan A. Mary', '8 Walnut St.', 'Malden', 'MA', 02148),
    -> (2661, 'Mixon Leatha', '100 School St.', 'Brookline', 'MA', 02146),
    -> (2890, 'Melane Sandy', '30 Case Rd.', 'Boston', 'MA', 02122),
    -> (3442, 'Novak Roanald', '42 Beacon St', 'Nashua', 'NH', 03060),
    -> (3566, 'Pierce Richard', '70 Park ST', 'Brookline', 'MA', 02146),
    -> (4022, 'Prior Lorrain', '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 (0.00 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> select * from Students;
+-------+-----------------+---------------+------------+-------+--------+
| st_no | name            | addr          | city       | state | zip_no |
+-------+-----------------+---------------+------------+-------+--------+
|  1011 | Edwards P. David | 10 Red Rd.    | Newton     | MA    | 2159   |
|  2415 | Grigan A. Mary   | 8 Walnut St.  | Malden     | MA    | 2148   |
|  2661 | Mixon Leatha     | 100 School St. | Brookline | MA    | 2146   |
|  2890 | Melane Sandy     | 30 Case Rd.   | Boston     | MA    | 2122   |
|  3442 | Novak Roanald    | 42 Beacon St  | Nashua     | NH    | 3060   |
|  3566 | Pierce Richard   | 70 Park ST    | Brookline  | MA    | 2146   |
|  4022 | Prior Lorrain    | 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:**

```
mysql> create table Instructors(
    -> emp_no INT PRIMARY KEY,
    -> name varchar (50),
    -> rank_title varchar (50),
    -> room_no varchar(20),
    -> tel_no varchar (30)
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> desc Instructors;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| emp_no     | int         | NO   | PRI | NULL    |       |
| name       | varchar(50) | YES  |     | NULL    |       |
| rank_title | varchar(50) | YES  |     | NULL    |       |
| room_no    | varchar(20) | YES  |     | NULL    |       |
| tel_no     | varchar(30) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Instructors (emp_no, name, rank_title, room_no, tel_no) 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', 'Assoc. Prof.', 90, 7024);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select * from Instructors;
+--------+--------------+--------------+---------+--------+
| emp_no | name         | rank_title   | room_no | tel_no |
+--------+--------------+--------------+---------+--------+
|     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  | Assoc. Prof. | 90      | 7024   |
+--------+--------------+--------------+---------+--------+
5 rows in set (0.00 sec)
```

◆**Courses:**

```
mysql> CREATE TABLE Courses (
    ->     c_no CHAR(10) PRIMARY KEY,  -- Specify the length of the CHAR type
    ->     name VARCHAR(50),
    ->     cr_no VARCHAR(15),
    ->     cap_no VARCHAR(20)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> desc Courses;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| c_no   | char(10)    | NO   | PRI | NULL    |       |
| name   | varchar(50) | YES  |     | NULL    |       |
| cr_no  | varchar(15) | YES  |     | NULL    |       |
| cap_no | varchar(20) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Courses (c_no, name, cr_no, cap_no) VALUES
    -> ('cn110', 'Introduction to Computing', 4, 120),
    -> ('cn210', 'Computer Programming', 4, 100),
    -> ('cn240', 'Computer Architecture', 3, 100),
    -> ('cn310', 'Data Structure', 3, 60),
    -> ('cn350', 'Higher Level Language', 3, 50),
    -> ('cn410', 'Software Engineering', 3, 40),
    -> ('cn460', 'Graphics', 3, 30);
Query OK, 7 rows affected (0.00 sec)
Records: 7  Duplicates: 0  Warnings: 0

mysql> select * from Courses;
+-------+--------------------------+-------+--------+
| c_no  | name                     | cr_no | cap_no |
+-------+--------------------------+-------+--------+
| cn110 | Introduction to Computing | 4     | 120    |
| cn210 | Computer Programming      | 4     | 100    |
| cn240 | Computer Architecture     | 3     | 100    |
| cn310 | Data Structure            | 3     | 60     |
| cn350 | Higher Level Language     | 3     | 50     |
| cn410 | Software Engineering      | 3     | 40     |
| cn460 | Graphics                  | 3     | 30     |
+-------+--------------------------+-------+--------+
7 rows in set (0.00 sec)
```

◆**Grades:**

```
mysql> CREATE TABLE Grades (
    ->     st_no INT NOT NULL,
    ->     emp_no INT NOT NULL,
    ->     c_no char(10) NOT NULL,
    ->     sem VARCHAR(10),
    ->     year varchar (20),
    ->     grade varchar (20),
    ->     PRIMARY KEY (st_no, c_no, sem, year),
    ->     FOREIGN KEY (st_no) REFERENCES Students(st_no),
    ->     FOREIGN KEY (emp_no) REFERENCES Instructors(emp_no),
    ->     FOREIGN KEY (c_no) REFERENCES Courses(c_no)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Grades;
+--------+------------+------+-----+---------+-------+
| Field  | Type       | Null | Key | Default | Extra |
+--------+------------+------+-----+---------+-------+
| st_no  | int        | NO   | PRI | NULL    |       |
| emp_no | int        | NO   | MUL | NULL    |       |
| c_no   | char(10)   | NO   | PRI | NULL    |       |
| sem    | varchar(10) | NO   | PRI | NULL    |       |
| year   | varchar(20) | NO   | PRI | NULL    |       |
| grade  | varchar(20) | YES  |     | NULL    |       |
+--------+------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Grades (st_no, emp_no, c_no, sem, year, grade) VALUES
    -> (1011, 019, 'cn110', 'Fall', 2001, 40),
    -> (2661, 019, 'cn110', 'Fall', 2001, 80),
    -> (3566, 019, 'cn110', 'Fall', 2001, 95),
    -> (5544, 019, 'cn110', 'Fall', 2001, 100),
    -> (1011, 023, 'cn110', 'Spring', 2002, 75),
    -> (4022, 023, 'cn110', 'Spring', 2002, 60),
    -> (3566, 019, 'cn240', 'Spring', 2002, 100),
    -> (5571, 019, 'cn240', 'Spring', 2002, 50),
    -> (2415, 019, 'cn240', 'Spring', 2002, 100),
    -> (3442, 234, 'cn410', 'Spring', 2002, 60),
    -> (5571, 234, 'cn410', 'Spring', 2002, 80),
    -> (1011, 019, 'cn210', 'Fall', 2003, 90),
    -> (2661, 019, 'cn210', 'Fall', 2003, 70),
    -> (3566, 019, 'cn210', 'Fall', 2003, 90),
    -> (5571, 019, 'cn210', 'Spring', 2003, 85),
    -> (4022, 019, 'cn210', 'Spring', 2003, 70),
    -> (5544, 056, 'cn240', 'Spring', 2003, 70),
    -> (1011, 056, 'cn240', 'Spring', 2003, 90),
    -> (4022, 056, 'cn240', 'Spring', 2003, 80),
    -> (2661, 234, 'cn310', 'Spring', 2003, 100),
    -> (4022, 234, 'cn310', 'Spring', 2003, 75);
Query OK, 21 rows affected (0.01 sec)
Records: 21  Duplicates: 0  Warnings: 0
```

```
mysql> select * from grades
    -> ;
+-------+--------+-------+--------+------+-------+
| st_no | emp_no | c_no  | sem    | year | grade |
+-------+--------+-------+--------+------+-------+
|  1011 |     19 | cn110 | Fall   | 2001 | 40    |
|  1011 |     23 | cn110 | Spring | 2002 | 75    |
|  1011 |     19 | cn210 | Fall   | 2003 | 90    |
|  1011 |     56 | cn240 | Spring | 2003 | 90    |
|  2415 |     19 | cn240 | Spring | 2002 | 100   |
|  2661 |     19 | cn110 | Fall   | 2001 | 80    |
|  2661 |     19 | cn210 | Fall   | 2003 | 70    |
|  2661 |    234 | cn310 | Spring | 2003 | 100   |
|  3442 |    234 | cn410 | Spring | 2002 | 60    |
|  3566 |     19 | cn110 | Fall   | 2001 | 95    |
|  3566 |     19 | cn210 | Fall   | 2003 | 90    |
|  3566 |     19 | cn240 | Spring | 2002 | 100   |
|  4022 |     23 | cn110 | Spring | 2002 | 60    |
|  4022 |     19 | cn210 | Spring | 2003 | 70    |
|  4022 |     56 | cn240 | Spring | 2003 | 80    |
|  4022 |    234 | cn310 | Spring | 2003 | 75    |
|  5544 |     19 | cn110 | Fall   | 2001 | 100   |
|  5544 |     56 | cn240 | Spring | 2003 | 70    |
|  5571 |     19 | cn210 | Spring | 2003 | 85    |
|  5571 |     19 | cn240 | Spring | 2002 | 50    |
|  5571 |    234 | cn410 | Spring | 2002 | 80    |
+-------+--------+-------+--------+------+-------+
21 rows in set (0.00 sec)
```

◆**Advising**

```
mysql> CREATE TABLE Advising (
    ->     st_no INT NOT NULL,
    ->     emp_no INT NOT NULL,
    ->     PRIMARY KEY (st_no, emp_no),
    ->     FOREIGN KEY (st_no) REFERENCES Students(st_no),
    ->     FOREIGN KEY (emp_no) REFERENCES Instructors(emp_no)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> desc Advising;
+--------+------+------+-----+---------+-------+
| Field  | Type | Null | Key | Default | Extra |
+--------+------+------+-----+---------+-------+
| st_no  | int  | NO   | PRI | NULL    |       |
| emp_no | int  | NO   | PRI | NULL    |       |
+--------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Advising (st_no, emp_no) VALUES
    -> (1011, 019),
    -> (2415, 019),
    -> (2661, 023),
    -> (2890, 023),
    -> (3442, 056),
    -> (3566, 126),
    -> (4022, 234),
    -> (5544, 023),
    -> (5571, 234);
Query OK, 9 rows affected (0.02 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> select * from Advising;
+-------+--------+
| st_no | emp_no |
+-------+--------+
|  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)
```

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

```
mysql> SELECT DISTINCT S.name
    -> FROM Students S, Grades G, Courses C
    -> WHERE S.st_no = G.st_no AND G.c_no = C.c_no AND C.cr_no = '4';
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Mixon Leatha     |
| Pierce Richard   |
| Prior Lorrain    |
| Rawlings Jerry   |
| Lewis Jerry      |
+------------------+
6 rows in set (0.04 sec)
```

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

```
mysql> SELECT st_no
    -> FROM Students
    -> WHERE NOT EXISTS (
    ->     SELECT * FROM Courses C
    ->     WHERE C.cr_no = '4'
    ->     AND NOT EXISTS (
    ->         SELECT * FROM Grades G
    ->         WHERE G.st_no = Students.st_no AND G.c_no = C.c_no
    ->     )
    -> );
+-------+
| st_no |
+-------+
|  1011 |
|  2661 |
|  3566 |
|  4022 |
+-------+
4 rows in set (0.00 sec)
```

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

```
mysql> SELECT DISTINCT S.name
    -> FROM Students S, Advising A, Grades G
    -> WHERE S.st_no = A.st_no AND A.emp_no = G.emp_no AND S.st_no = G.st_no;
+------------------+
| name             |
+------------------+
| Edwards P. David |
| Grigan A. Mary   |
| Prior Lorrain    |
| Lewis Jerry      |
+------------------+
4 rows in set (0.00 sec)
```

4. Find the names of students who took cs210 and cs310.

```
mysql> SELECT DISTINCT S.name
    -> FROM Students S
    -> WHERE S.st_no IN (SELECT G1.st_no FROM Grades G1 WHERE G1.c_no = 'cn210')
    ->   AND S.st_no IN (SELECT G2.st_no FROM Grades G2 WHERE G2.c_no = 'cn310');
+---------------+
| name          |
+---------------+
| Mixon Leatha  |
| Prior Lorrain |
+---------------+
2 rows in set (0.00 sec)
```

5. Find the names of all students whose advisor is not a full professor.

```
mysql> SELECT DISTINCT S.name
    -> FROM Students S, Advising A, Instructors I
    -> WHERE S.st_no = A.st_no AND A.emp_no = I.emp_no AND I.rank_title != 'Professor';
+---------------+
| name          |
+---------------+
| Novak Roanald |
| Pierce Richard |
| Prior Lorrain |
| 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 I1.name
    -> FROM Instructors I1, Grades G, Advising A, Instructors I2
    -> WHERE G.st_no = A.st_no AND G.emp_no = I1.emp_no AND A.emp_no = I2.emp_no AND I1.room_no = I2.room_no AND I1.emp_no != I2.emp_no;
+--------------+
| name         |
+--------------+
| Will Samuel  |
| Exxon George |
+--------------+
2 rows in set (0.00 sec)
```

7. Find course numbers for courses that enroll exactly two students;

```
mysql> SELECT c_no
    -> FROM Grades
    -> GROUP BY c_no
    -> HAVING COUNT(DISTINCT st_no) = 2;
+-------+
| c_no  |
+-------+
| cn310 |
| cn410 |
+-------+
2 rows in set (0.00 sec)
```

8. Find the names of all students for whom no other student lives in the same city.

```
mysql> SELECT S1.name
    -> FROM Students S1
    -> WHERE NOT EXISTS (
    ->     SELECT * FROM Students S2
    ->     WHERE S1.city = S2.city AND S1.st_no != S2.st_no
    -> );
+-------------------+
| name              |
+-------------------+
| Edwards P. David  |
| Grigan A. Mary    |
| Novak Roanald     |
| Lewis Jerry       |
+-------------------+
4 rows in set (0.04 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 G.c_no
    -> FROM Grades G, Students S, Instructors I
    -> WHERE G.st_no = S.st_no AND G.emp_no = I.emp_no AND S.city = 'Boston' AND I.rank_title = 'Assoc. Prof.';
+-------+
| c_no  |
+-------+
| cn310 |
| cn240 |
+-------+
2 rows 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.tel_no
    -> FROM Instructors I, Grades G, Students S
    -> WHERE G.emp_no = I.emp_no AND G.st_no = S.st_no AND S.city = 'Boston';
+--------+
| tel_no |
+--------+
| 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 S1.name
    -> FROM Students S1
    -> WHERE NOT EXISTS (
    ->     SELECT G.c_no
    ->     FROM Grades G, Students S2
    ->     WHERE S2.name = 'Richard Pierce' AND S2.st_no = G.st_no
    ->     AND NOT EXISTS (
    ->         SELECT * FROM Grades G1 WHERE G1.st_no = S1.st_no AND G1.c_no = G.c_no
    ->     )
    -> );
+-----------------+
| name            |
+-----------------+
| Edwards P. David |
| Grigan A. Mary   |
| Mixon Leatha     |
| Melane Sandy     |
| Novak Roanald    |
| Pierce Richard   |
| Prior Lorrain    |
| Rawlings Jerry   |
| Lewis Jerry      |
+-----------------+
9 rows in set (0.00 sec)
```

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

```
mysql> SELECT S.name
    -> FROM Students S, Grades G
    -> WHERE S.st_no = G.st_no
    -> GROUP BY S.st_no, S.name
    -> HAVING COUNT(G.c_no) = 1;
+-----------------+
| name            |
+-----------------+
| Grigan A. Mary |
| Novak Roanald  |
+-----------------+
2 rows in set (0.04 sec)
```

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

```
mysql> SELECT I.name
    -> FROM Instructors I
    -> WHERE I.emp_no NOT IN (SELECT DISTINCT G.emp_no FROM Grades G);
+---------------+
| 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.name
    -> FROM Instructors I, Grades G
    -> WHERE I.emp_no = G.emp_no AND G.sem = 'Spring' AND G.year = '2001'
    -> GROUP BY I.name
    -> HAVING COUNT(DISTINCT G.c_no) = 1;
Empty set (0.04 sec)
```