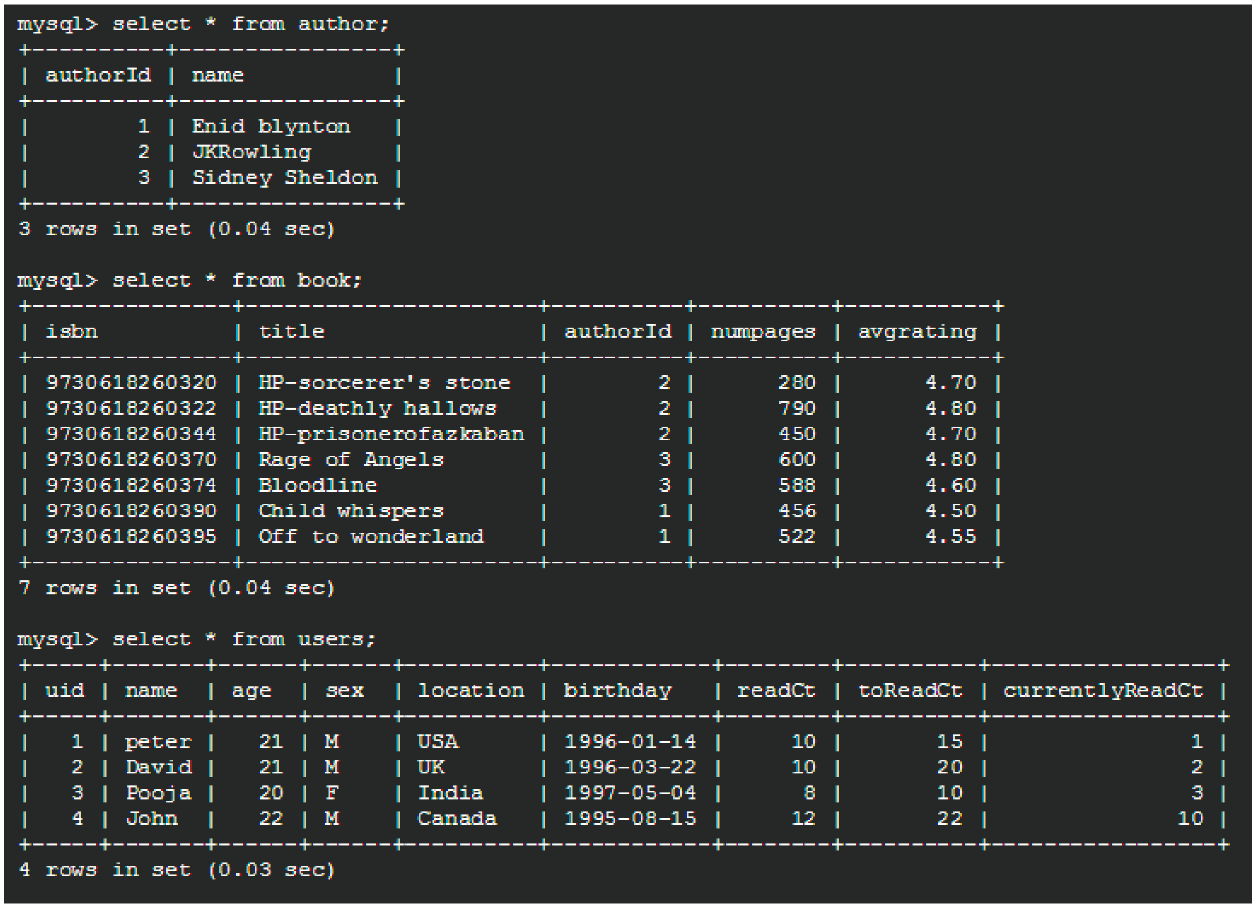
STUDENT ID : 8654990832

NAME : RADHIKA RAVICHANDRAN

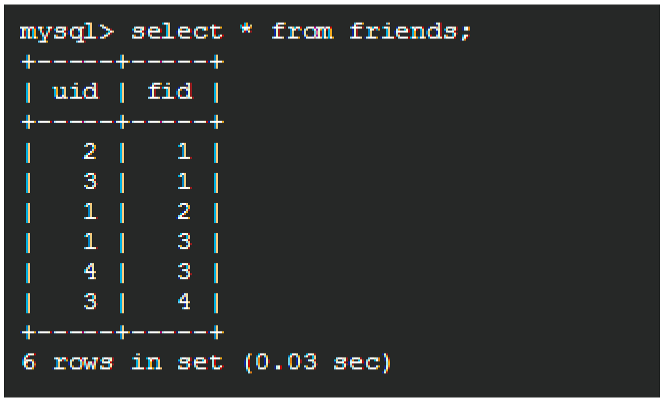
LANGUAGE USED : MySQL

SET 1: GoodReads database

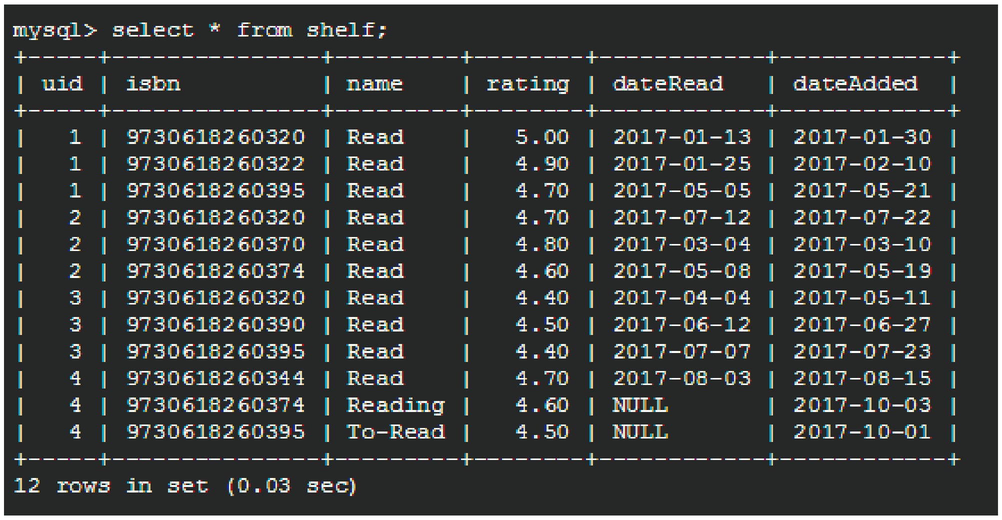
Tables created for GoodReads database



```
mysql> select * from author;
+----------+----------------+
| authorId | name           |
+----------+----------------+
|        1 | Enid blynton   |
|        2 | JKRowling      |
|        3 | Sidney Sheldon |
+----------+----------------+
3 rows in set (0.04 sec)

mysql> select * from book;
+---------------+----------------------+----------+----------+-----------+
| isbn          | title                | authorId | numpages | avgrating |
+---------------+----------------------+----------+----------+-----------+
| 9730618260320 | HP-sorcerer's stone  |        2 |      280 |      4.70 |
| 9730618260322 | HP-deathly hallows   |        2 |      790 |      4.80 |
| 9730618260344 | HP-prisonerofazkaban |        2 |      450 |      4.70 |
| 9730618260370 | Rage of Angels       |        3 |      600 |      4.80 |
| 9730618260374 | Bloodline            |        3 |      588 |      4.60 |
| 9730618260390 | Child whispers       |        1 |      456 |      4.50 |
| 9730618260395 | Off to wonderland    |        1 |      522 |      4.55 |
+---------------+----------------------+----------+----------+-----------+
7 rows in set (0.04 sec)

mysql> select * from users;
+-----+-------+------+------+----------+------------+--------+----------+-----------------+
| uid | name  | age  | sex  | location | birthday   | readCt | toReadCt | currentlyReadCt |
+-----+-------+------+------+----------+------------+--------+----------+-----------------+
|   1 | peter |   21 | M    | USA      | 1996-01-14 |     10 |       15 |               1 |
|   2 | David |   21 | M    | UK       | 1996-03-22 |     10 |       20 |               2 |
|   3 | Pooja |   20 | F    | India    | 1997-05-04 |      8 |       10 |               3 |
|   4 | John  |   22 | M    | Canada   | 1995-08-15 |     12 |       22 |              10 |
+-----+-------+------+------+----------+------------+--------+----------+-----------------+
4 rows in set (0.03 sec)
```



```
mysql> select * from friends;
+-----+-----+
| uid | fid |
+-----+-----+
|   2 |   1 |
|   3 |   1 |
|   1 |   2 |
|   1 |   3 |
|   4 |   3 |
|   3 |   4 |
+-----+-----+
6 rows in set (0.03 sec)
```



```
mysql> select * from shelf;
+-----+---------------+---------+--------+------------+------------+
| uid | isbn          | name    | rating | dateRead   | dateAdded  |
+-----+---------------+---------+--------+------------+------------+
|   1 | 9730618260320 | Read    |   5.00 | 2017-01-13 | 2017-01-30 |
|   1 | 9730618260322 | Read    |   4.90 | 2017-01-25 | 2017-02-10 |
|   1 | 9730618260395 | Read    |   4.70 | 2017-05-05 | 2017-05-21 |
|   2 | 9730618260320 | Read    |   4.70 | 2017-07-12 | 2017-07-22 |
|   2 | 9730618260370 | Read    |   4.80 | 2017-03-04 | 2017-03-10 |
|   2 | 9730618260374 | Read    |   4.60 | 2017-05-08 | 2017-05-19 |
|   3 | 9730618260320 | Read    |   4.40 | 2017-04-04 | 2017-05-11 |
|   3 | 9730618260390 | Read    |   4.50 | 2017-06-12 | 2017-06-27 |
|   3 | 9730618260395 | Read    |   4.40 | 2017-07-07 | 2017-07-23 |
|   4 | 9730618260344 | Read    |   4.70 | 2017-08-03 | 2017-08-15 |
|   4 | 9730618260374 | Reading |   4.60 | NULL       | 2017-10-03 |
|   4 | 9730618260395 | To-Read |   4.50 | NULL       | 2017-10-01 |
+-----+---------------+---------+--------+------------+------------+
12 rows in set (0.03 sec)
```

1. User adds a new book to his shelf with a rating. Update the average rating of that book.
SQL Query:

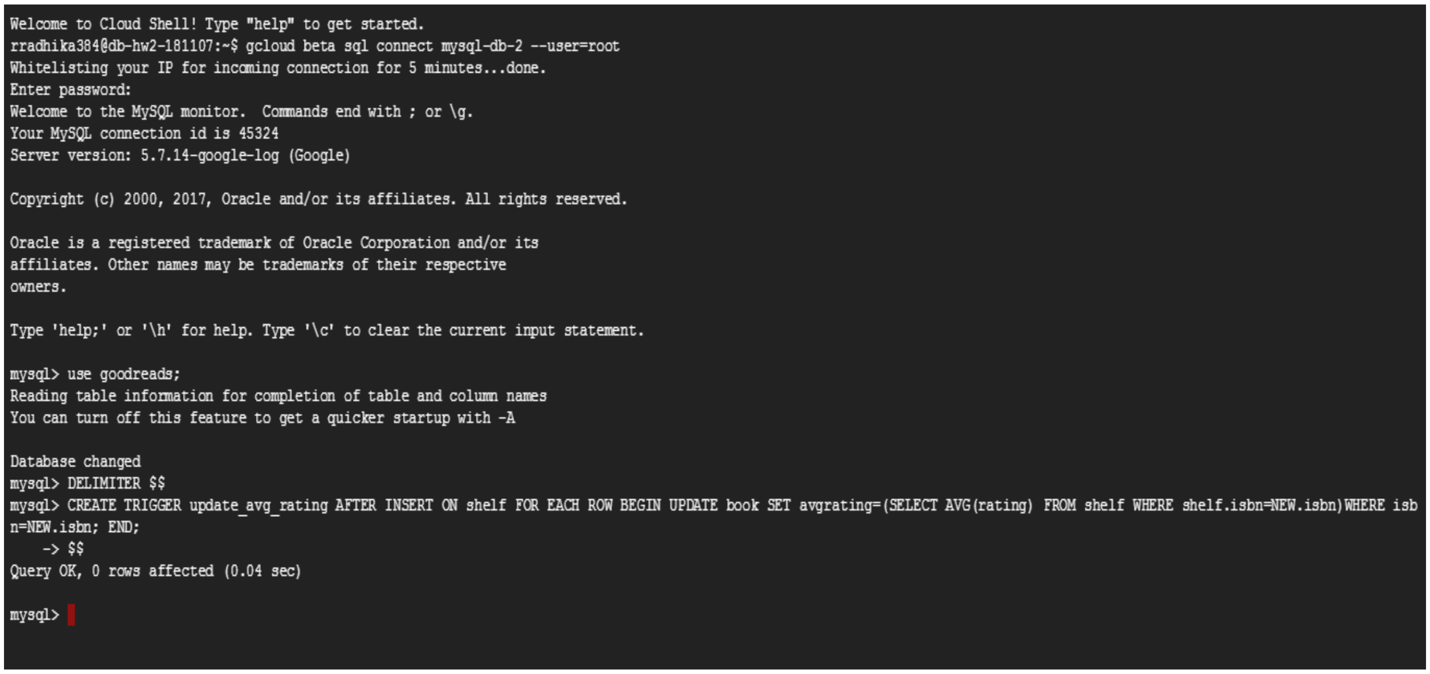
DELIMITER $$

CREATE TRIGGER update_avg_rating AFTER INSERT ON shelf FOR EACH ROW BEGIN UPDATE book SET avgrating=(SELECT AVG(rating) FROM shelf WHERE shelf.isbn=NEW.isbn)WHERE isbn=NEW.isbn;

END;

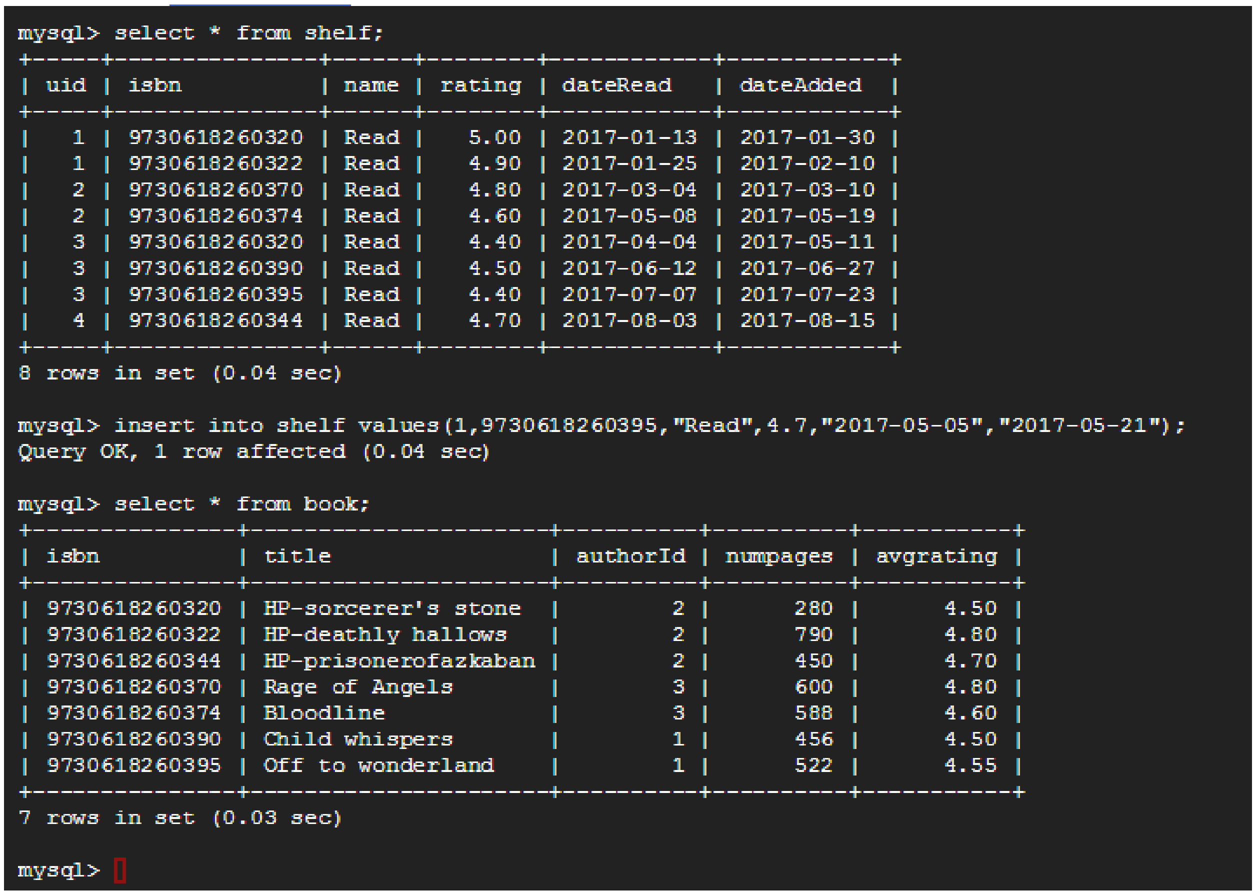
$$

Screenshot:



```
Welcome to Cloud Shell! Type "help" to get started.
rradhika384@db-hw2-181107:~$ gcloud beta sql connect mysql-db-2 --user=root
Whitelisting your IP for incoming connection for 5 minutes...done.
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 45324
Server version: 5.7.14-google-log (Google)

Copyright (c) 2000, 2017, 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> use goodreads;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> DELIMITER $$
mysql> CREATE TRIGGER update_avg_rating AFTER INSERT ON shelf FOR EACH ROW BEGIN UPDATE book SET avgrating=(SELECT AVG(rating) FROM shelf WHERE shelf.isbn=NEW.isbn)WHERE isb
n=NEW.isbn; END;
    -> $$
Query OK, 0 rows affected (0.04 sec)

mysql>
```

After inserting new book with new rating in shelf table, the trigger updates the average rating of the book in book table



```
mysql> select * from shelf;
+-----+---------------+------+--------+------------+------------+
| uid | isbn          | name | rating | dateRead   | dateAdded  |
+-----+---------------+------+--------+------------+------------+
|   1 | 9730618260320 | Read |   5.00 | 2017-01-13 | 2017-01-30 |
|   1 | 9730618260322 | Read |   4.90 | 2017-01-25 | 2017-02-10 |
|   2 | 9730618260370 | Read |   4.80 | 2017-03-04 | 2017-03-10 |
|   2 | 9730618260374 | Read |   4.60 | 2017-05-08 | 2017-05-19 |
|   3 | 9730618260320 | Read |   4.40 | 2017-04-04 | 2017-05-11 |
|   3 | 9730618260390 | Read |   4.50 | 2017-06-12 | 2017-06-27 |
|   3 | 9730618260395 | Read |   4.40 | 2017-07-07 | 2017-07-23 |
|   4 | 9730618260344 | Read |   4.70 | 2017-08-03 | 2017-08-15 |
+-----+---------------+------+--------+------------+------------+
8 rows in set (0.04 sec)

mysql> insert into shelf values(1,9730618260395,"Read",4.7,"2017-05-05","2017-05-21");
Query OK, 1 row affected (0.04 sec)

mysql> select * from book;
+---------------+----------------------+----------+----------+-----------+
| isbn          | title                | authorId | numpages | avgrating |
+---------------+----------------------+----------+----------+-----------+
| 9730618260320 | HP-sorcerer's stone  |        2 |      280 |      4.50 |
| 9730618260322 | HP-deathly hallows   |        2 |      790 |      4.80 |
| 9730618260344 | HP-prisonerofazkaban |        2 |      450 |      4.70 |
| 9730618260370 | Rage of Angels       |        3 |      600 |      4.80 |
| 9730618260374 | Bloodline            |        3 |      588 |      4.60 |
| 9730618260390 | Child whispers       |        1 |      456 |      4.50 |
| 9730618260395 | Off to wonderland    |        1 |      522 |      4.55 |
+---------------+----------------------+----------+----------+-----------+
7 rows in set (0.03 sec)

mysql>
```

Explanation:

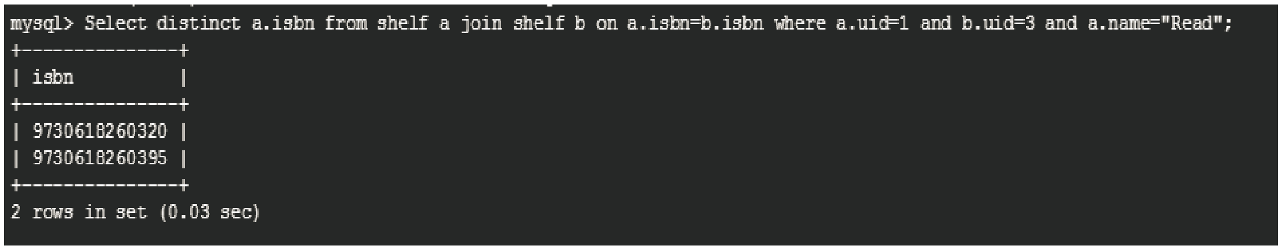
Since the average rating of the book must be updated after user adds book to shelf, I have created a trigger which will update the average rating of the book by checking new book's isbn with isbn already present in book table. The total average of all ratings given for the particular book will be constantly updated.

2. Find the names of the common books that were read by any two users X and Y.

SQL Query:

Select distinct a.isbn from shelf a join shelf b on a.isbn=b.isbn where a.uid=1 and b.uid=3 and a.name=”Read”;

Screenshot:



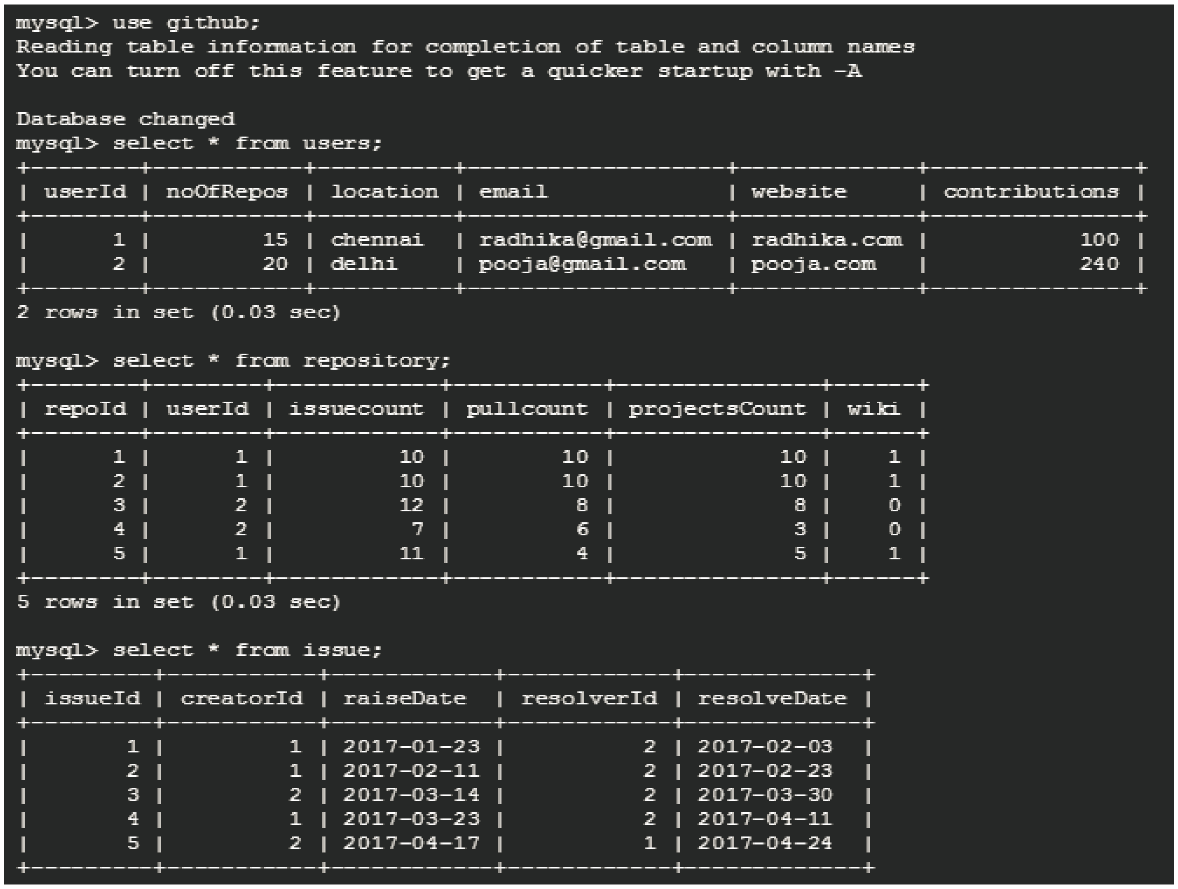
```
mysql> Select distinct a.isbn from shelf a join shelf b on a.isbn=b.isbn where a.uid=1 and b.uid=3 and a.name="Read";
+---------------+
| isbn          |
+---------------+
| 9730618260320 |
| 9730618260395 |
+---------------+
2 rows in set (0.03 sec)
```

Explanation:

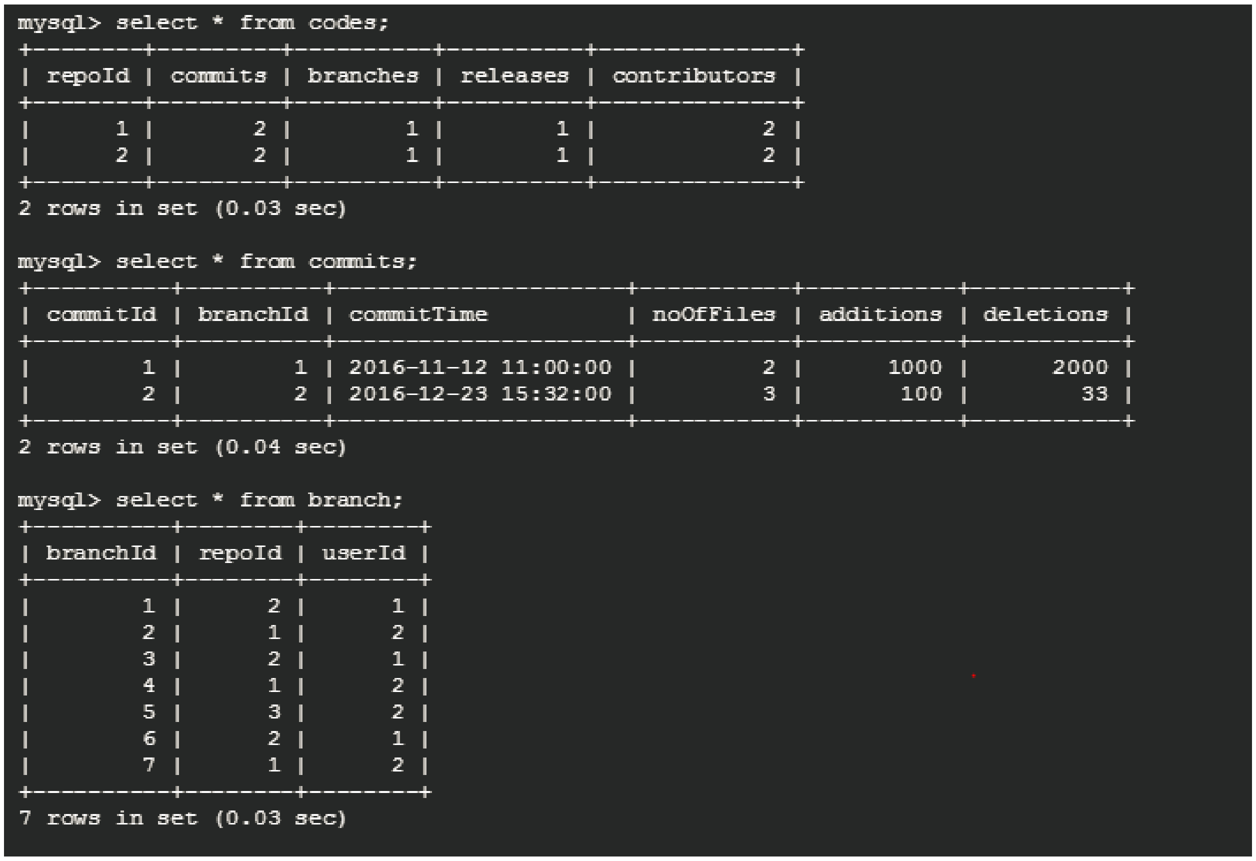
Since the common books read by 2 users must be listed,I have hardcoded the two user id's.Here user X is uid 1 and user Y is uid 3.And only the books which have been read should be listed so I have included the condition where name must be "Read"

SET 2: GitHub Database

Tables created for Github database



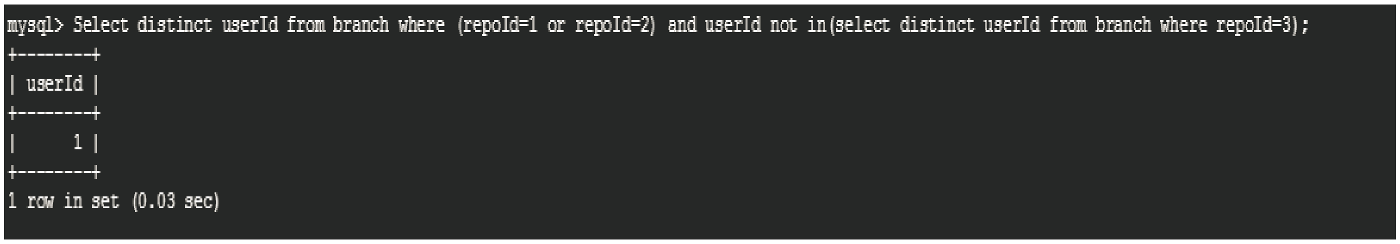
```
mysql> use github;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from users;
+--------+-----------+----------+-------------------+-------------+---------------+
| userId | noOfRepos | location | email             | website     | contributions |
+--------+-----------+----------+-------------------+-------------+---------------+
|      1 |        15 | chennai  | radhika@gmail.com | radhika.com |           100 |
|      2 |        20 | delhi    | pooja@gmail.com   | pooja.com   |           240 |
+--------+-----------+----------+-------------------+-------------+---------------+
2 rows in set (0.03 sec)

mysql> select * from repository;
+--------+--------+------------+-----------+---------------+------+
| repoId | userId | issuecount | pullcount | projectsCount | wiki |
+--------+--------+------------+-----------+---------------+------+
|      1 |      1 |         10 |        10 |            10 |    1 |
|      2 |      1 |         10 |        10 |            10 |    1 |
|      3 |      2 |         12 |         8 |             8 |    0 |
|      4 |      2 |          7 |         6 |             3 |    0 |
|      5 |      1 |         11 |         4 |             5 |    1 |
+--------+--------+------------+-----------+---------------+------+
5 rows in set (0.03 sec)

mysql> select * from issue;
+---------+-----------+------------+------------+-------------+
| issueId | creatorId | raiseDate  | resolverId | resolveDate |
+---------+-----------+------------+------------+-------------+
|       1 |         1 | 2017-01-23 |          2 | 2017-02-03  |
|       2 |         1 | 2017-02-11 |          2 | 2017-02-23  |
|       3 |         2 | 2017-03-14 |          2 | 2017-03-30  |
|       4 |         1 | 2017-03-23 |          2 | 2017-04-11  |
|       5 |         2 | 2017-04-17 |          1 | 2017-04-24  |
+---------+-----------+------------+------------+-------------+
```



```
mysql> select * from codes;
+--------+---------+----------+----------+--------------+
| repoId | commits | branches | releases | contributors |
+--------+---------+----------+----------+--------------+
|      1 |       2 |        1 |        1 |            2 |
|      2 |       2 |        1 |        1 |            2 |
+--------+---------+----------+----------+--------------+
2 rows in set (0.03 sec)

mysql> select * from commits;
+----------+----------+---------------------+-----------+-----------+-----------+
| commitId | branchId | commitTime          | noOfFiles | additions | deletions |
+----------+----------+---------------------+-----------+-----------+-----------+
|        1 |        1 | 2016-11-12 11:00:00 |         2 |      1000 |      2000 |
|        2 |        2 | 2016-12-23 15:32:00 |         3 |       100 |        33 |
+----------+----------+---------------------+-----------+-----------+-----------+
2 rows in set (0.04 sec)

mysql> select * from branch;
+----------+--------+--------+
| branchId | repoId | userId |
+----------+--------+--------+
|        1 |      2 |      1 |
|        2 |      1 |      2 |
|        3 |      2 |      1 |
|        4 |      1 |      2 |
|        5 |      3 |      2 |
|        6 |      2 |      1 |
|        7 |      1 |      2 |
+----------+--------+--------+
7 rows in set (0.03 sec)
```

1.Find the users who made branches of either of repositories X or Y but not of a repository Z.
SQL Query:
Select distinct userId from branch where (repoId=1 or repoId=2) and userId not in(select distinct userId from branch where repoId=3);

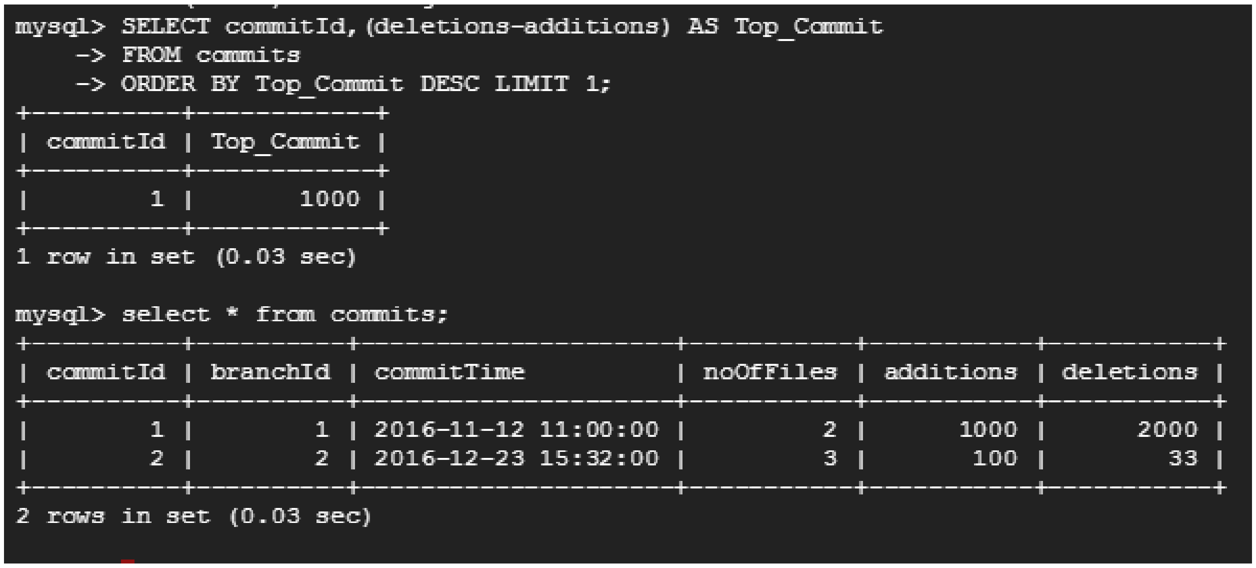
Screenshot:



```
mysql> Select distinct userId from branch where (repoId=1 or repoId=2) and userId not in(select distinct userId from branch where repoId=3);
+--------+
| userId |
+--------+
|      1 |
+--------+
1 row in set (0.03 sec)
```

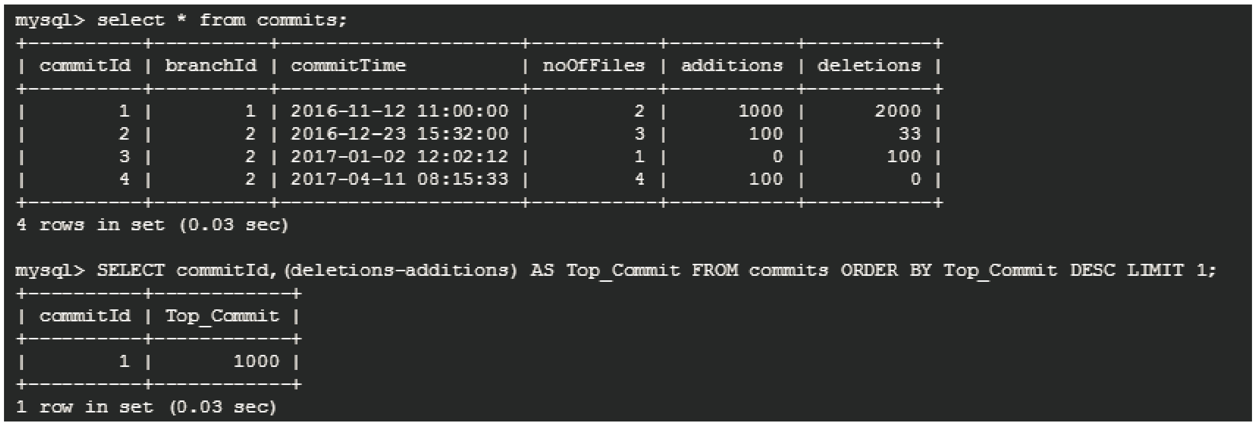
Explanation:

As it is mentioned that only the users which have branches of repositories X or Y but not of Z should be listed,I have hardcoded the repository values : where repoId=1 is X,repoId=2 is Y and repoId=3 is Z.
Since branchId is primary key in branch table,I have used NOT IN to filter out the users which do not have branches for repository 3 but have branches for repositories 1 OR 2.

2. Find the top commit with the highest lines of code reduced. (Hint: We need to find the maximized value of: number of deletions - number of additions in each commit).
SQL Query:
SELECT commitId,(deletions-additions) AS Top_Commit FROM commits ORDER BY Top_Commit DESC LIMIT 1;
Screenshot:


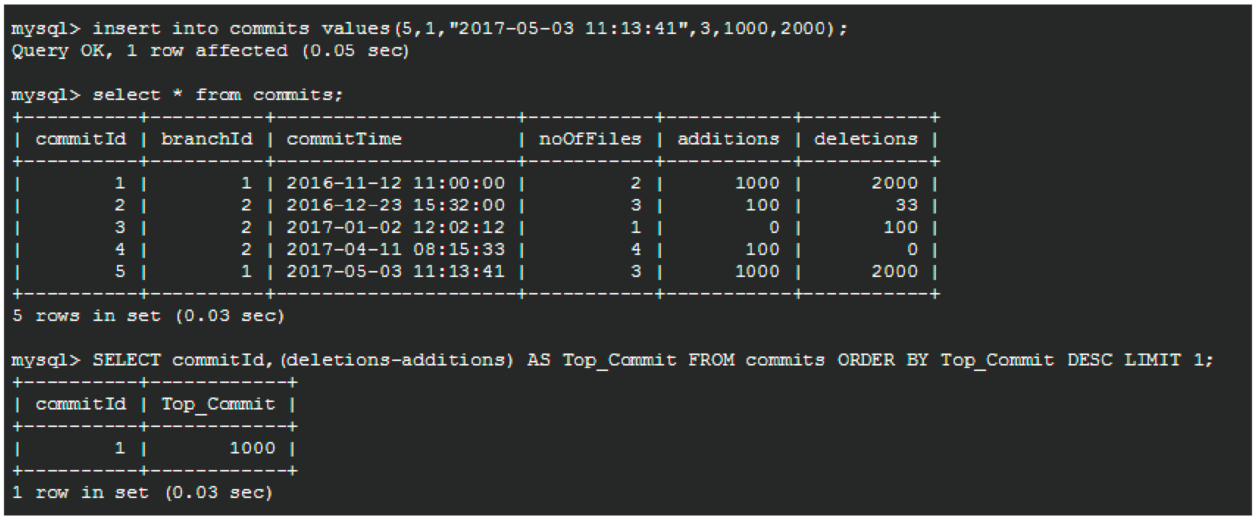
```
mysql> SELECT commitId,(deletions-additions) AS Top_Commit
    -> FROM commits
    -> ORDER BY Top_Commit DESC LIMIT 1;
+----------+------------+
| commitId | Top_Commit |
+----------+------------+
|        1 |       1000 |
+----------+------------+
1 row in set (0.03 sec)

mysql> select * from commits;
+----------+----------+---------------------+-----------+-----------+-----------+
| commitId | branchId | commitTime          | noOfFiles | additions | deletions |
+----------+----------+---------------------+-----------+-----------+-----------+
|        1 |        1 | 2016-11-12 11:00:00 |         2 |      1000 |      2000 |
|        2 |        2 | 2016-12-23 15:32:00 |         3 |       100 |        33 |
+----------+----------+---------------------+-----------+-----------+-----------+
2 rows in set (0.03 sec)
```

Explanation:
As mentioned in the instructions the TOP COMMIT is the commitId with Maximum(number of deletions-number of additions),I have found the difference between deletions and additions and have sorted them in descending order and retrieved the top 1 commit.

Boundary conditions like when there are 0 additions and 100 deletions and 0 deletions and 100 additions will also be taken care of.


```
mysql> select * from commits;
+----------+----------+---------------------+-----------+-----------+-----------+
| commitId | branchId | commitTime          | noOfFiles | additions | deletions |
+----------+----------+---------------------+-----------+-----------+-----------+
|        1 |        1 | 2016-11-12 11:00:00 |         2 |      1000 |      2000 |
|        2 |        2 | 2016-12-23 15:32:00 |         3 |       100 |        33 |
|        3 |        2 | 2017-01-02 12:02:12 |         1 |         0 |       100 |
|        4 |        2 | 2017-04-11 08:15:33 |         4 |       100 |         0 |
+----------+----------+---------------------+-----------+-----------+-----------+
4 rows in set (0.03 sec)

mysql> SELECT commitId,(deletions-additions) AS Top_Commit FROM commits ORDER BY Top_Commit DESC LIMIT 1;
+----------+------------+
| commitId | Top_Commit |
+----------+------------+
|        1 |       1000 |
+----------+------------+
1 row in set (0.03 sec)
```

And when there are 2 rows with same difference value between number of deletions and number of additions then one of the top commits will be displayed. Please find below the screenshot for the same.

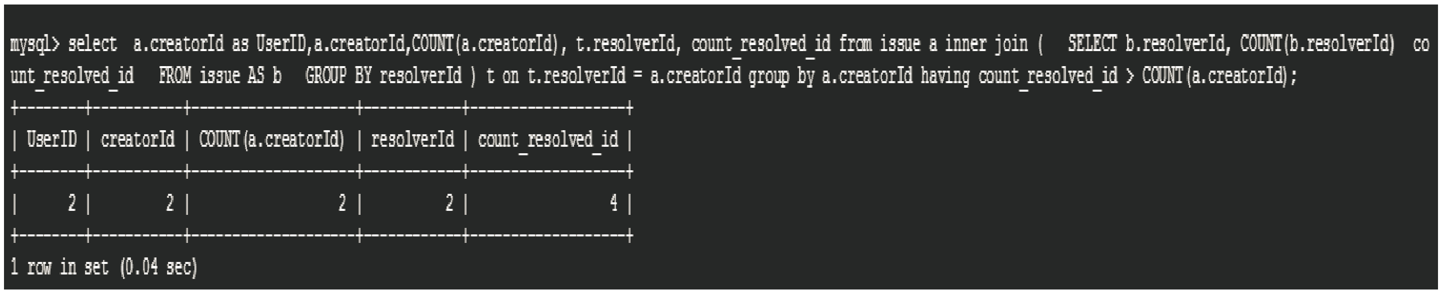


```
mysql> insert into commits values(5,1,"2017-05-03 11:13:41",3,1000,2000);
Query OK, 1 row affected (0.05 sec)

mysql> select * from commits;
+----------+----------+---------------------+-----------+-----------+-----------+
| commitId | branchId | commitTime          | noOfFiles | additions | deletions |
+----------+----------+---------------------+-----------+-----------+-----------+
|        1 |        1 | 2016-11-12 11:00:00 |         2 |      1000 |      2000 |
|        2 |        2 | 2016-12-23 15:32:00 |         3 |       100 |        33 |
|        3 |        2 | 2017-01-02 12:02:12 |         1 |         0 |       100 |
|        4 |        2 | 2017-04-11 08:15:33 |         4 |       100 |         0 |
|        5 |        1 | 2017-05-03 11:13:41 |         3 |      1000 |      2000 |
+----------+----------+---------------------+-----------+-----------+-----------+
5 rows in set (0.03 sec)

mysql> SELECT commitId,(deletions-additions) AS Top_Commit FROM commits ORDER BY Top_Commit DESC LIMIT 1;
+----------+------------+
| commitId | Top_Commit |
+----------+------------+
|        1 |       1000 |
+----------+------------+
1 row in set (0.03 sec)
```

3. (BONUS question) List the users who solved more issues than they raised. (i.e. number of issues in which they were the resolver is greater than the number of issues where they were the creator.)
SQL Query:

Select a.creatorId as UserID,a.creatorId,COUNT(a.creatorId),t.resolverId,count_resolved_id from issue a inner join(Select b.resolverId,COUNT(b.resolverId) count_resolved_id FROM issue AS b GROUP BY resolverId) t on t.resolverId=a.creatorId group by a.creatorId having count_resolved_id>COUNT(a.creatorId);

Screenshot:



```
mysql> select  a.creatorId as UserID,a.creatorId,COUNT(a.creatorId), t.resolverId, count_resolved_id from issue a inner join (   SELECT b.resolverId, COUNT(b.resolverId)  co
unt_resolved_id   FROM issue AS b   GROUP BY resolverId ) t on t.resolverId = a.creatorId group by a.creatorId having count_resolved_id > COUNT(a.creatorId);
+--------+-----------+--------------------+------------+-------------------+
| UserID | creatorId | COUNT(a.creatorId) | resolverId | count_resolved_id |
+--------+-----------+--------------------+------------+-------------------+
|      2 |         2 |                  2 |          2 |                 4 |
+--------+-----------+--------------------+------------+-------------------+
1 row in set (0.04 sec)
```

DDL Commands:

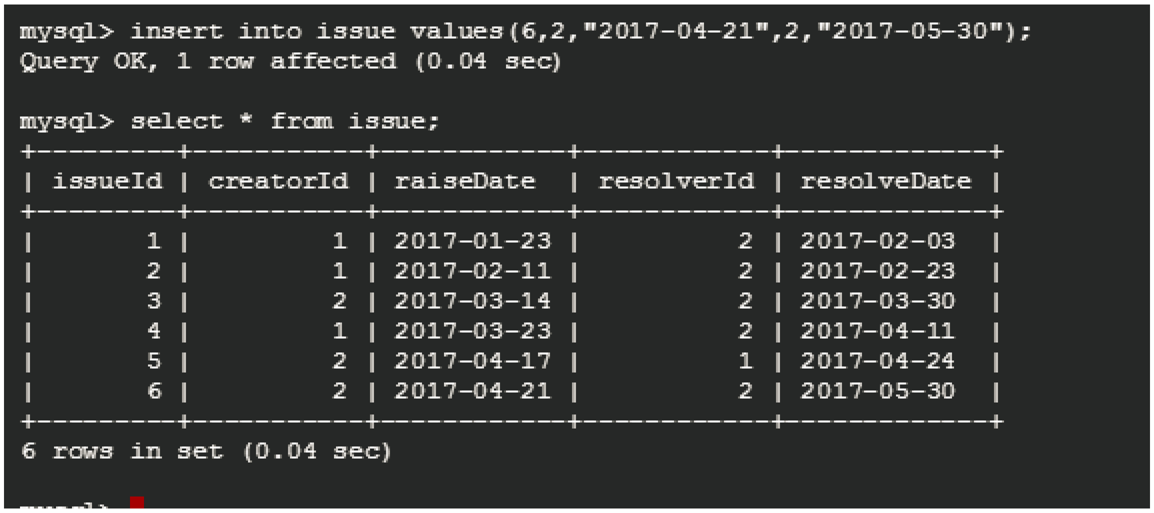
Table creation:

Create table issue (issueId int, creatorId int not null, raiseDate date, resolverId int, resolveDate date, primary key (issueId), constraint fk2 foreign key(creatorId) references users(userId), constraint fk3 foreign key(resolverId) references users(userId));

Table insertion:

Insert into issue values(6,2,"2017-04-21",2,"2017-05-30");

Please find below screenshots for DDL commands



```
mysql> insert into issue values(6,2,"2017-04-21",2,"2017-05-30");
Query OK, 1 row affected (0.04 sec)

mysql> select * from issue;
+---------+-----------+------------+------------+-------------+
| issueId | creatorId | raiseDate  | resolverId | resolveDate |
+---------+-----------+------------+------------+-------------+
|       1 |         1 | 2017-01-23 |          2 | 2017-02-03  |
|       2 |         1 | 2017-02-11 |          2 | 2017-02-23  |
|       3 |         2 | 2017-03-14 |          2 | 2017-03-30  |
|       4 |         1 | 2017-03-23 |          2 | 2017-04-11  |
|       5 |         2 | 2017-04-17 |          1 | 2017-04-24  |
|       6 |         2 | 2017-04-21 |          2 | 2017-05-30  |
+---------+-----------+------------+------------+-------------+
6 rows in set (0.04 sec)
```

Explanation:

Grouped issues with same creatorId and resolverId and then obtained count of it. Then listed the userId which is the creatorId(or)resolverId   whose count of number of issues resolved is greater than no off issues created.