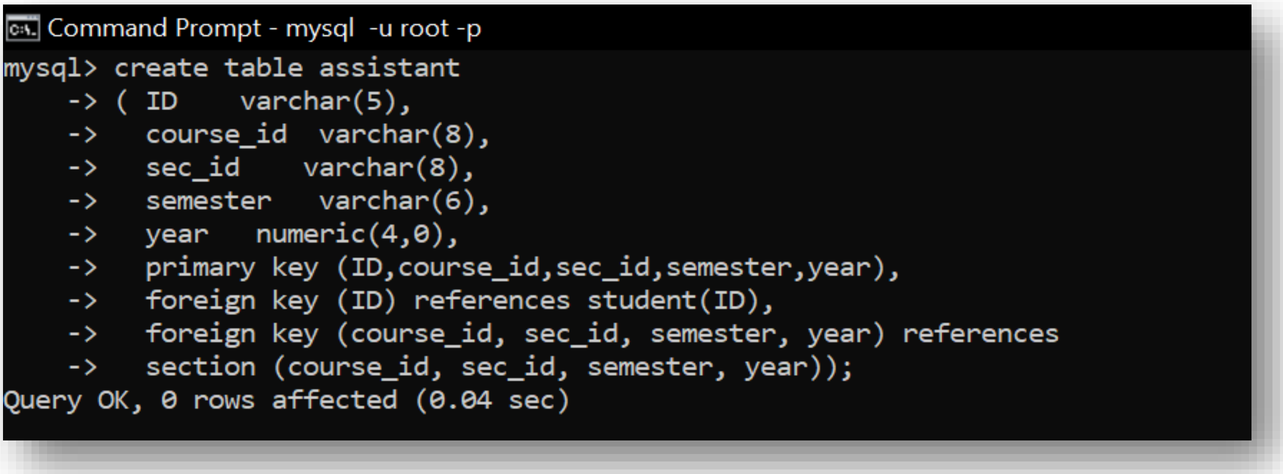
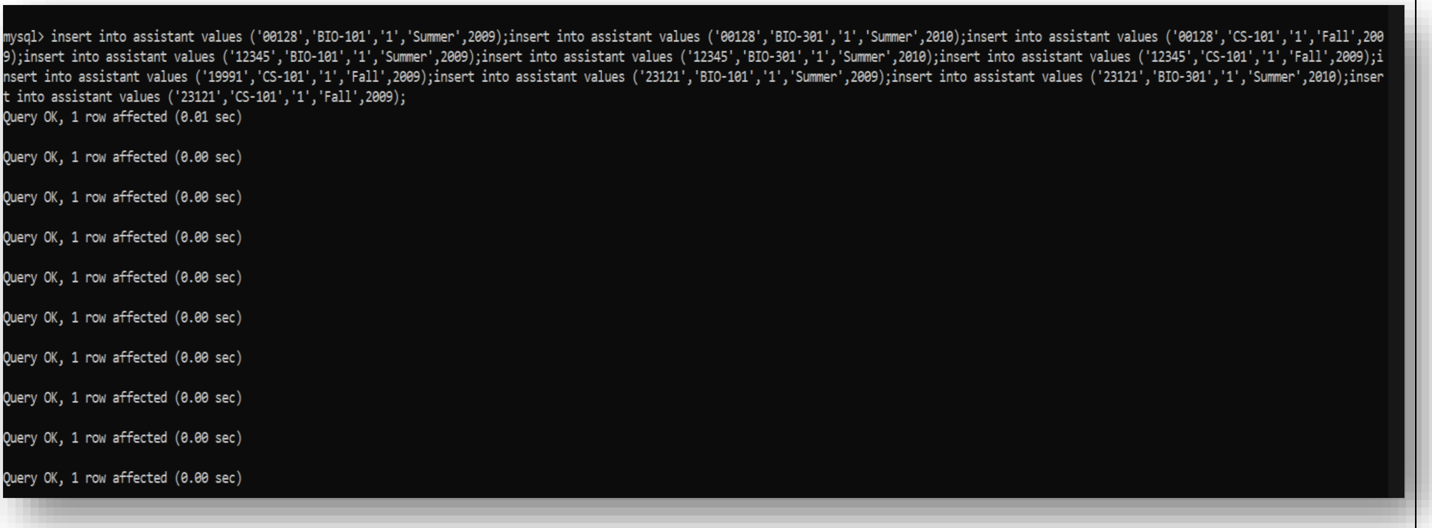
DBMS ASSIGNMENT 3

- Each offering of a course (i.e. a section) can have many Teaching assistants; each teaching assistant is a student. Extend the existing schema(Add/Alter tables) to accommodate this requirement.

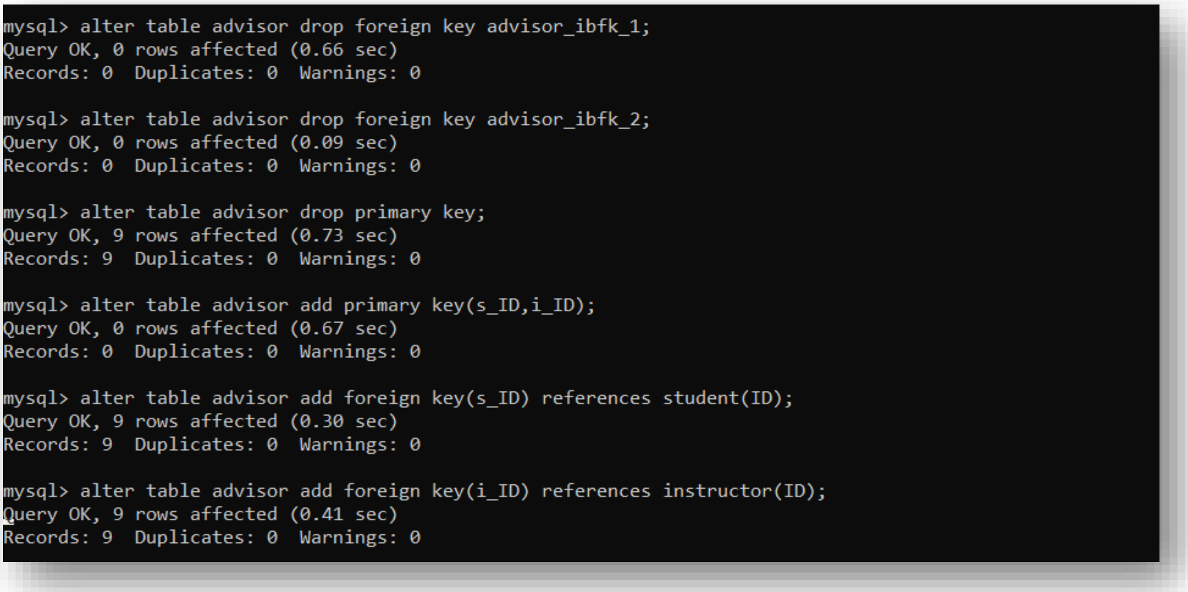


```
mysql> create table assistant
    -> ( ID    varchar(5),
    ->   course_id  varchar(8),
    ->   sec_id    varchar(8),
    ->   semester   varchar(6),
    ->   year   numeric(4,0),
    ->   primary key (ID,course_id,sec_id,semester,year),
    ->   foreign key (ID) references student(ID),
    ->   foreign key (course_id, sec_id, semester, year) references
    ->   section (course_id, sec_id, semester, year));
Query OK, 0 rows affected (0.04 sec)
```

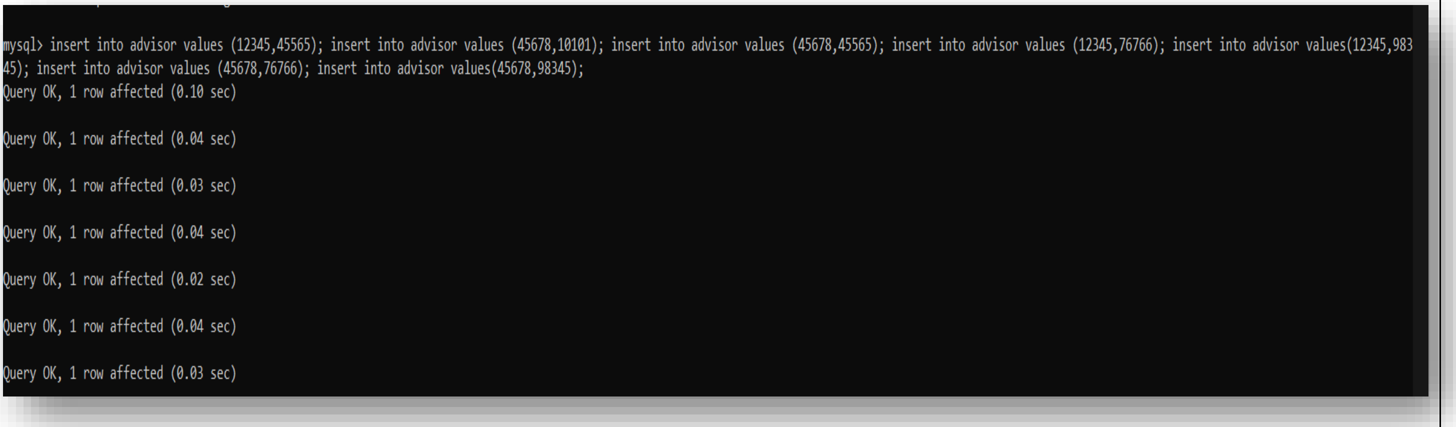


```
mysql> insert into assistant values ('00128','BIO-101','1','Summer',2009);insert into assistant values ('00128','BIO-301','1','Summer',2010);insert into assistant values ('00128','CS-101','1','Fall',200
9);insert into assistant values ('12345','BIO-101','1','Summer',2009);insert into assistant values ('12345','BIO-301','1','Summer',2010);insert into assistant values ('12345','CS-101','1','Fall',2009);i
nsert into assistant values ('19991','CS-101','1','Fall',2009);insert into assistant values ('23121','BIO-101','1','Summer',2009);insert into assistant values ('23121','BIO-301','1','Summer',2010);inser
t into assistant values ('23121','CS-101','1','Fall',2009);
Query OK, 1 row affected (0.01 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)
```

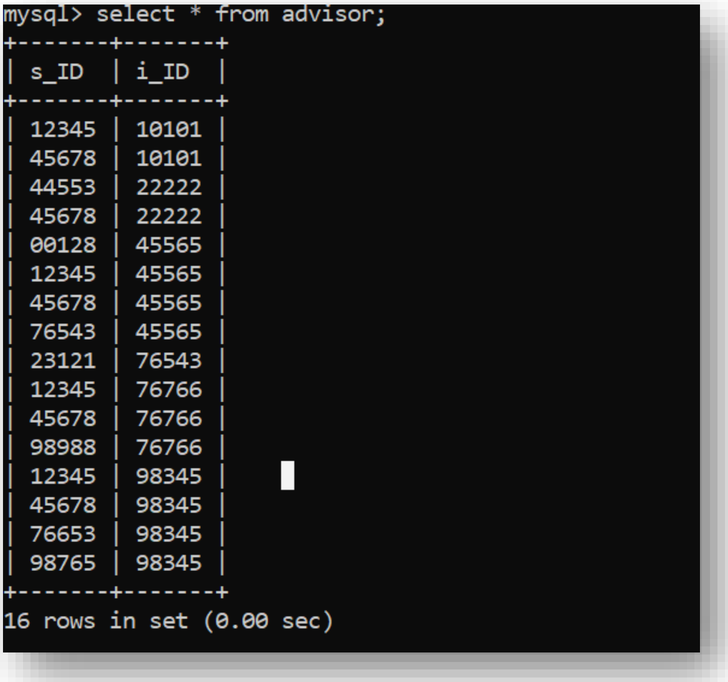
- Alter the schema to allow a student to have multiple advisors and make sure that you are able to insert multiple advisors for a student.



```
mysql> alter table advisor drop foreign key advisor_ibfk_1;
Query OK, 0 rows affected (0.66 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table advisor drop foreign key advisor_ibfk_2;
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table advisor drop primary key;
Query OK, 9 rows affected (0.73 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> alter table advisor add primary key(s_ID,i_ID);
Query OK, 0 rows affected (0.67 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table advisor add foreign key(s_ID) references student(ID);
Query OK, 9 rows affected (0.30 sec)
Records: 9  Duplicates: 0  Warnings: 0

mysql> alter table advisor add foreign key(i_ID) references instructor(ID);
Query OK, 9 rows affected (0.41 sec)
Records: 9  Duplicates: 0  Warnings: 0
```

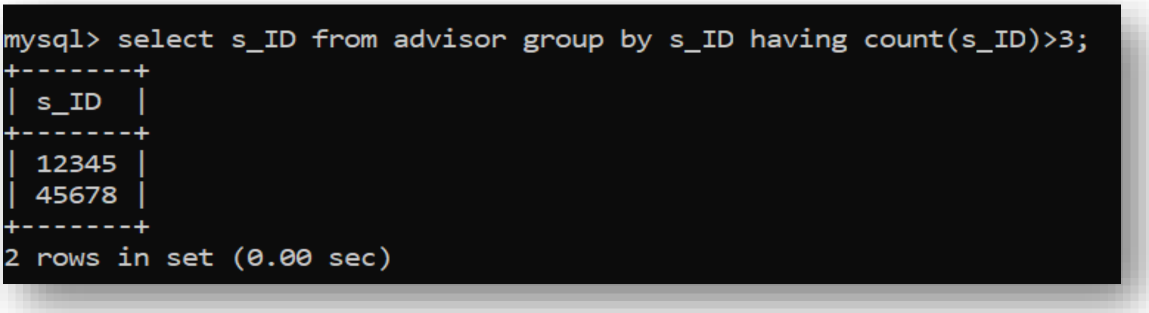


```
mysql> insert into advisor values (12345,45565); insert into advisor values (45678,10101); insert into advisor values (45678,45565); insert into advisor values (12345,76766); insert into advisor values(12345,983
45); insert into advisor values (45678,76766); insert into advisor values(45678,98345);
Query OK, 1 row affected (0.10 sec)

Query OK, 1 row affected (0.04 sec)

Query OK, 1 row affected (0.03 sec)

Query OK, 1 row affected (0.04 sec)

Query OK, 1 row affected (0.02 sec)

Query OK, 1 row affected (0.04 sec)

Query OK, 1 row affected (0.03 sec)
```



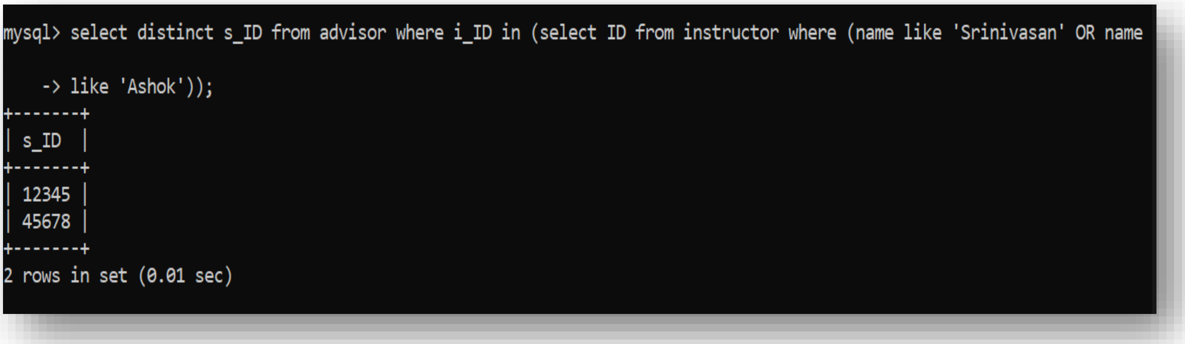
```
mysql> select * from advisor;
+-------+-------+
| s_ID  | i_ID  |
+-------+-------+
| 12345 | 10101 |
| 45678 | 10101 |
| 44553 | 22222 |
| 45678 | 22222 |
| 00128 | 45565 |
| 12345 | 45565 |
| 45678 | 45565 |
| 76543 | 45565 |
| 23121 | 76543 |
| 12345 | 76766 |
| 45678 | 76766 |
| 98988 | 76766 |
| 12345 | 98345 |
| 45678 | 98345 |
| 76653 | 98345 |
| 98765 | 98345 |
+-------+-------+
16 rows in set (0.00 sec)
```

- Find all students who have more than 3 advisors



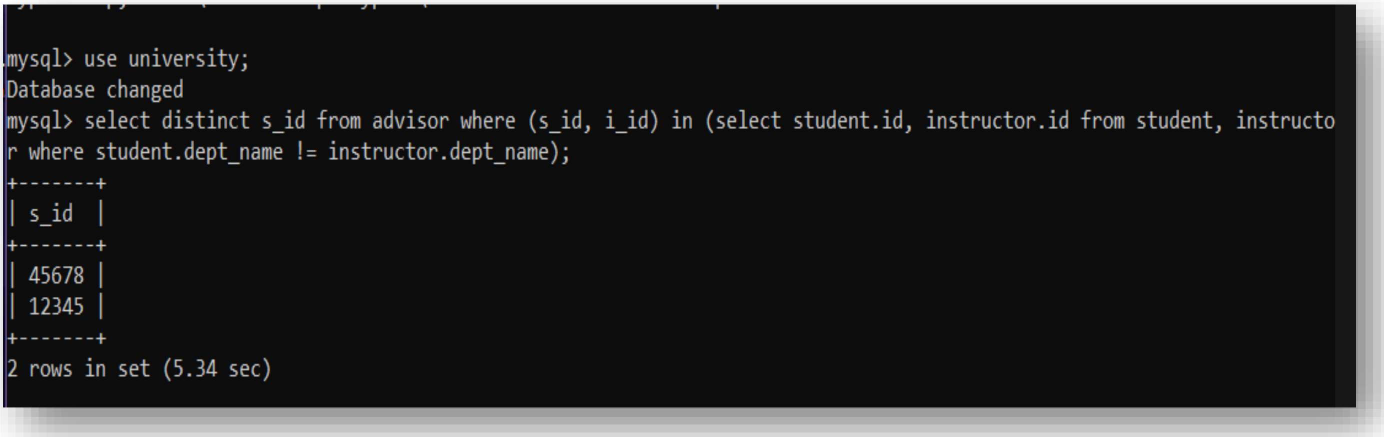
```
mysql> select s_ID from advisor group by s_ID having count(s_ID)>3;
+-------+
| s_ID  |
+-------+
| 12345 |
| 45678 |
+-------+
2 rows in set (0.00 sec)
```

- Find all students who are co-advised by Prof. Srinivas and Prof. Ashok.



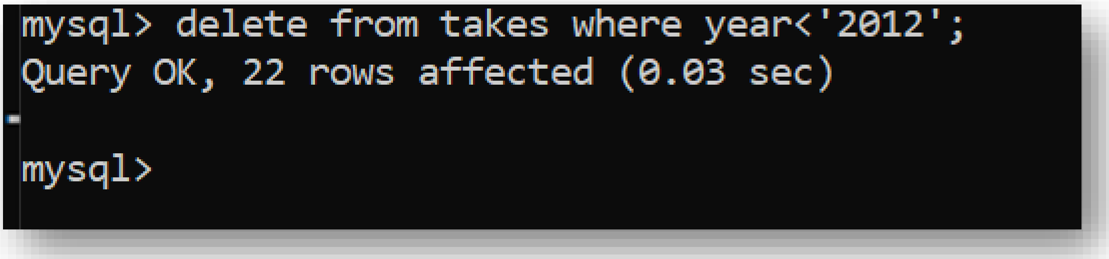
```
mysql> select distinct s_ID from advisor where i_ID in (select ID from instructor where (name like 'Srinivasan' OR name

    -> like 'Ashok'));
+-------+
| s_ID  |
+-------+
| 12345 |
| 45678 |
+-------+
2 rows in set (0.01 sec)
```

- Find students advised by instructors from different departments. etc.

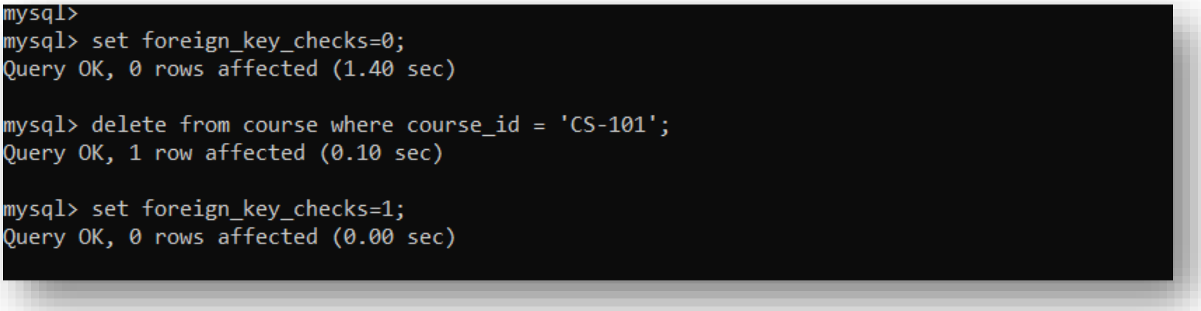


```
mysql> use university;
Database changed
mysql> select distinct s_id from advisor where (s_id, i_id) in (select student.id, instructor.id from student, instructo
r where student.dept_name != instructor.dept_name);
+-------+
| s_id  |
+-------+
| 45678 |
| 12345 |
+-------+
2 rows in set (5.34 sec)
```

- Delete all information in the database which is more than 10 years old. Add data as necessary to verify your query.



```
mysql> delete from takes where year<'2012';
Query OK, 22 rows affected (0.03 sec)

mysql>
```



```
mysql>
mysql> set foreign_key_checks=0;
Query OK, 0 rows affected (1.40 sec)

mysql> delete from course where course_id = 'CS-101';
Query OK, 1 row affected (0.10 sec)

mysql> set foreign_key_checks=1;
Query OK, 0 rows affected (0.00 sec)
```

- Delete the course CS 101. Any course which has CS 101 as a prereq should remove CS 101 from its prereq set. Create a cascade constraint to enforce the above rule, and verify that it is working.

drop table prereq;

create table prereq(course_id varchar(8), prereq_id varchar(8), primary key (course_id, prereq_id), foreign key (course_id) references course(course_id) on delete cascade, foreign key(prereq_id) references course(course_id) on delete cascade);

insert into prereq values ('BIO-301', 'BIO-101');

insert into prereq values ('BIO-399', 'BIO-101');

insert into prereq values ('CS-190', 'CS-101');

insert into prereq values ('CS-315', 'CS-101');

insert into prereq values ('CS-319', 'CS-101');

```
insert into prereq values ('CS-347', 'CS-101');

insert into prereq values ('EE-181', 'PHY-101');

delete from course where course_id = 'CS-101';
```