DEPARTMENT OF INFORMATION TECHNOLOGY

Database Systems
(IT252)

Mini Project

Institute Database

Team No: 2

Gaurang Jitendra Velingkar - 191IT113

Niraj Nandish - 191IT234

Ritvik Mahesh - 191IT246

Overview

We have implemented a database system for an institute that stores Student, Faculty, Department and Course details along with relationships between these tables. We have created 6 tables in total.

User requirements -

Users should be able to:

- The institute has one or more departments
- Each department has one HOD and offers one or more courses
- Each course will have zero or more students enrolled in it
- Every faculty belongs to a department and can teach one or more courses
- When a student enrolls into a course, calculate total number of enrolled students in a course
- Each student can be enrolled into one or more courses
- Store GPA of a student in a course and automatically calculate CGPA and total credits
- Generate statistics based on student grades
- If a student has just enrolled into a course, their GPA is 0.00 by default
- When a course is added, update number of courses offered by the department automatically

In summary, the user requirement is for a system that can store basic student, faculty, department and course information while also calculating grades and generating statistics involving relationships between any permutations of the tables to assess the quality of education being imparted by the institute.

Users who would use the system:

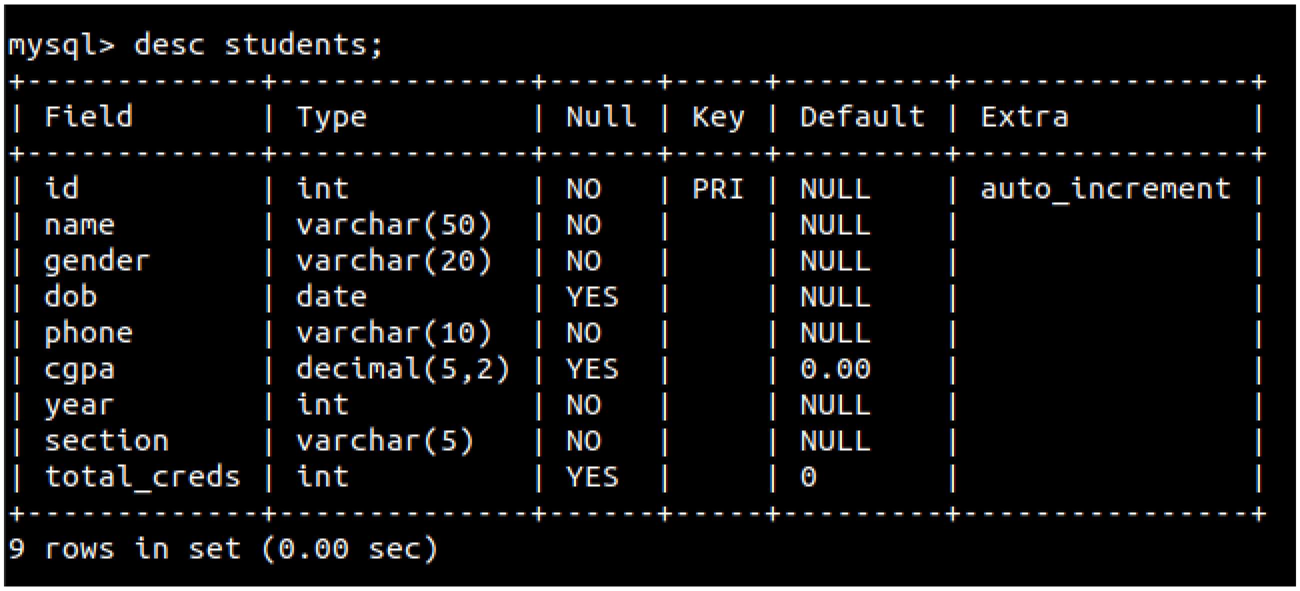
- Students
- Course Instructors
- Department Heads
- Other faculty

We have attached screenshots of the database schema along with the initial values inserted in the database below, along with a few queries that we performed.

Note - Since the queries were performed on three different systems, there may be some inconsistencies in data because we entered data individually to evaluate the outcome of our queries. even though we tried our best to keep the data as consistent as possible. The structure of tables, however, remains the same across all of our systems.

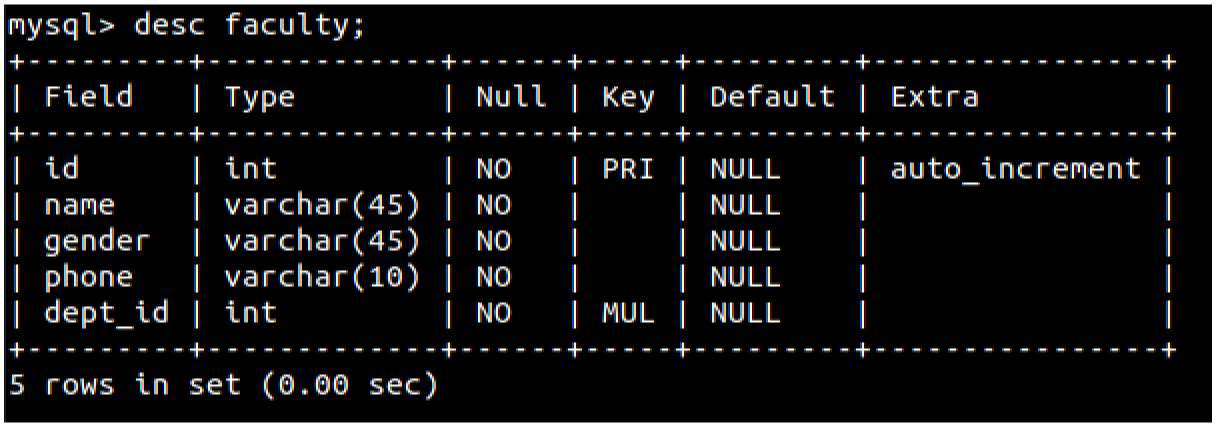
We have 6 tables as described below :-

1. *Students*
   Stores student details in columns as given below.



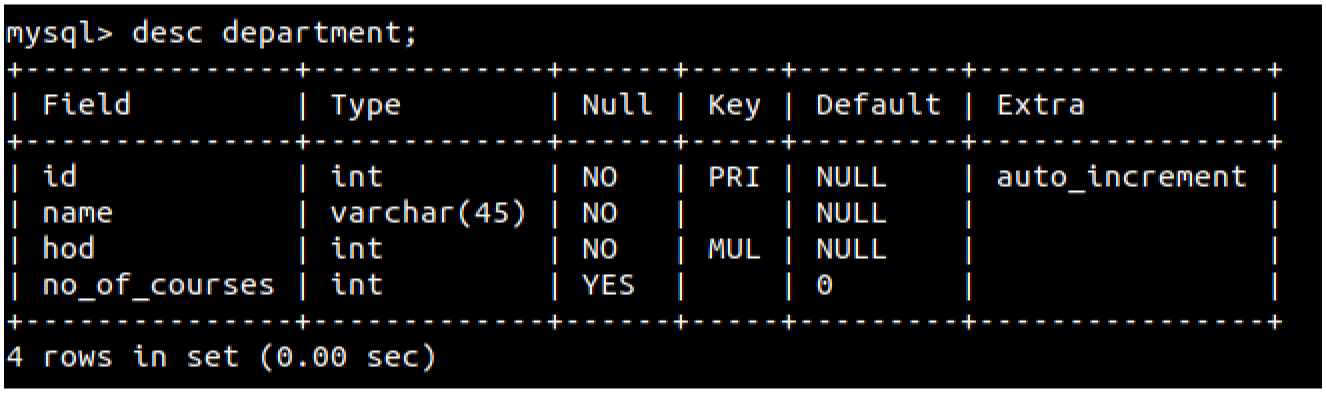
```
mysql> desc students;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| id          | int          | NO   | PRI | NULL    | auto_increment |
| name        | varchar(50)  | NO   |     | NULL    |                |
| gender      | varchar(20)  | NO   |     | NULL    |                |
| dob         | date         | YES  |     | NULL    |                |
| phone       | varchar(10)  | NO   |     | NULL    |                |
| cgpa        | decimal(5,2) | YES  |     | 0.00    |                |
| year        | int          | NO   |     | NULL    |                |
| section     | varchar(5)   | NO   |     | NULL    |                |
| total_creds | int          | YES  |     | 0       |                |
+-------------+--------------+------+-----+---------+----------------+
9 rows in set (0.00 sec)
```

2. *Faculty*
   Stores faculty details in columns as given below. `dept_id` is a foreign key to the Department table.



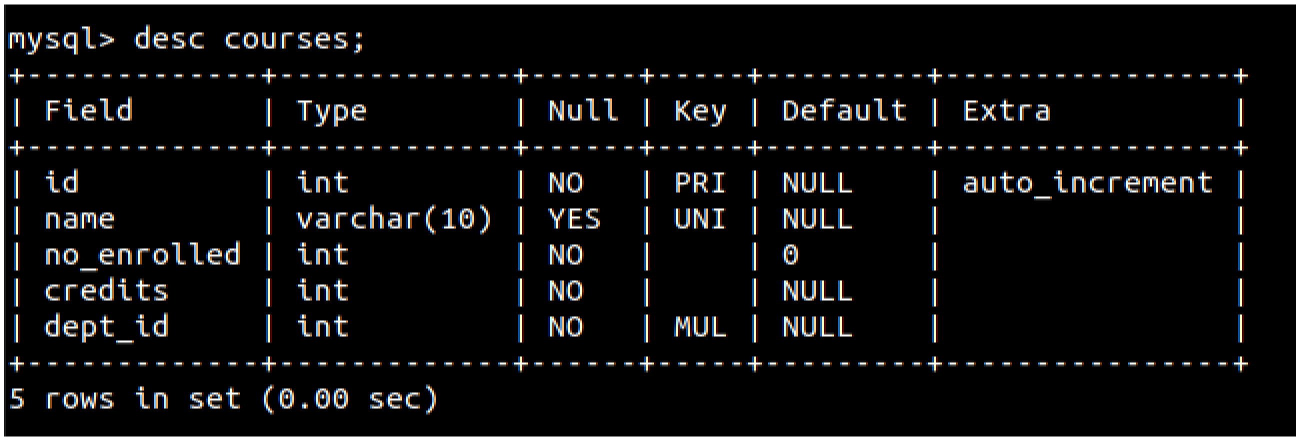
```
mysql> desc faculty;
+---------+-------------+------+-----+---------+----------------+
| Field   | Type        | Null | Key | Default | Extra          |
+---------+-------------+------+-----+---------+----------------+
| id      | int         | NO   | PRI | NULL    | auto_increment |
| name    | varchar(45) | NO   |     | NULL    |                |
| gender  | varchar(45) | NO   |     | NULL    |                |
| phone   | varchar(10) | NO   |     | NULL    |                |
| dept_id | int         | NO   | MUL | NULL    |                |
+---------+-------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

3. *Department*
   Stores department details in columns as given below. `hod` is a foreign key to the Faculty table, and stores the ID of the department's HoD.



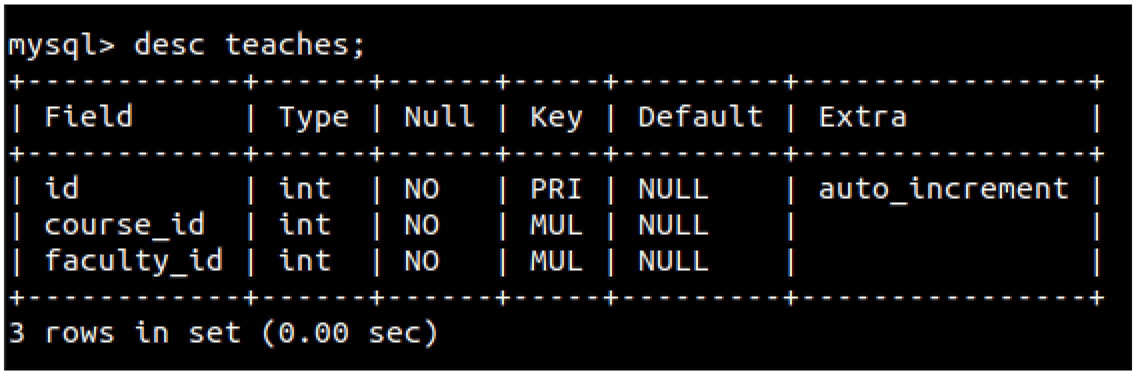
```
mysql> desc department;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| id            | int         | NO   | PRI | NULL    | auto_increment |
| name          | varchar(45) | NO   |     | NULL    |                |
| hod           | int         | NO   | MUL | NULL    |                |
| no_of_courses | int         | YES  |     | 0       |                |
+---------------+-------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

4. *Courses*
   Stores course details in columns as given below. `dept_id` is a foreign key to the Department table. Course name is unique.



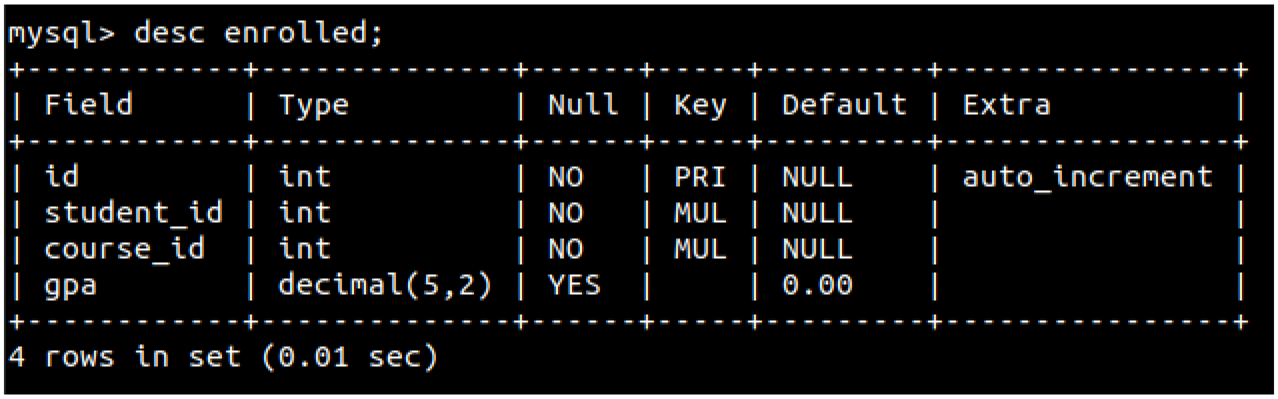
```
mysql> desc courses;
+-------------+-------------+------+-----+---------+----------------+
| Field       | Type        | Null | Key | Default | Extra          |
+-------------+-------------+------+-----+---------+----------------+
| id          | int         | NO   | PRI | NULL    | auto_increment |
| name        | varchar(10) | YES  | UNI | NULL    |                |
| no_enrolled | int         | NO   |     | 0       |                |
| credits     | int         | NO   |     | NULL    |                |
| dept_id     | int         | NO   | MUL | NULL    |                |
+-------------+-------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

5. *Teaches*
   Stores relations between the Faculty table and Courses table. It stores the details of courses that a faculty teaches. `faculty_id` is a foreign key to the Faculty table. `course_id` is a foreign key to the Courses table.



```
mysql> desc teaches;
+------------+------+------+-----+---------+----------------+
| Field      | Type | Null | Key | Default | Extra          |
+------------+------+------+-----+---------+----------------+
| id         | int  | NO   | PRI | NULL    | auto_increment |
| course_id  | int  | NO   | MUL | NULL    |                |
| faculty_id | int  | NO   | MUL | NULL    |                |
+------------+------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

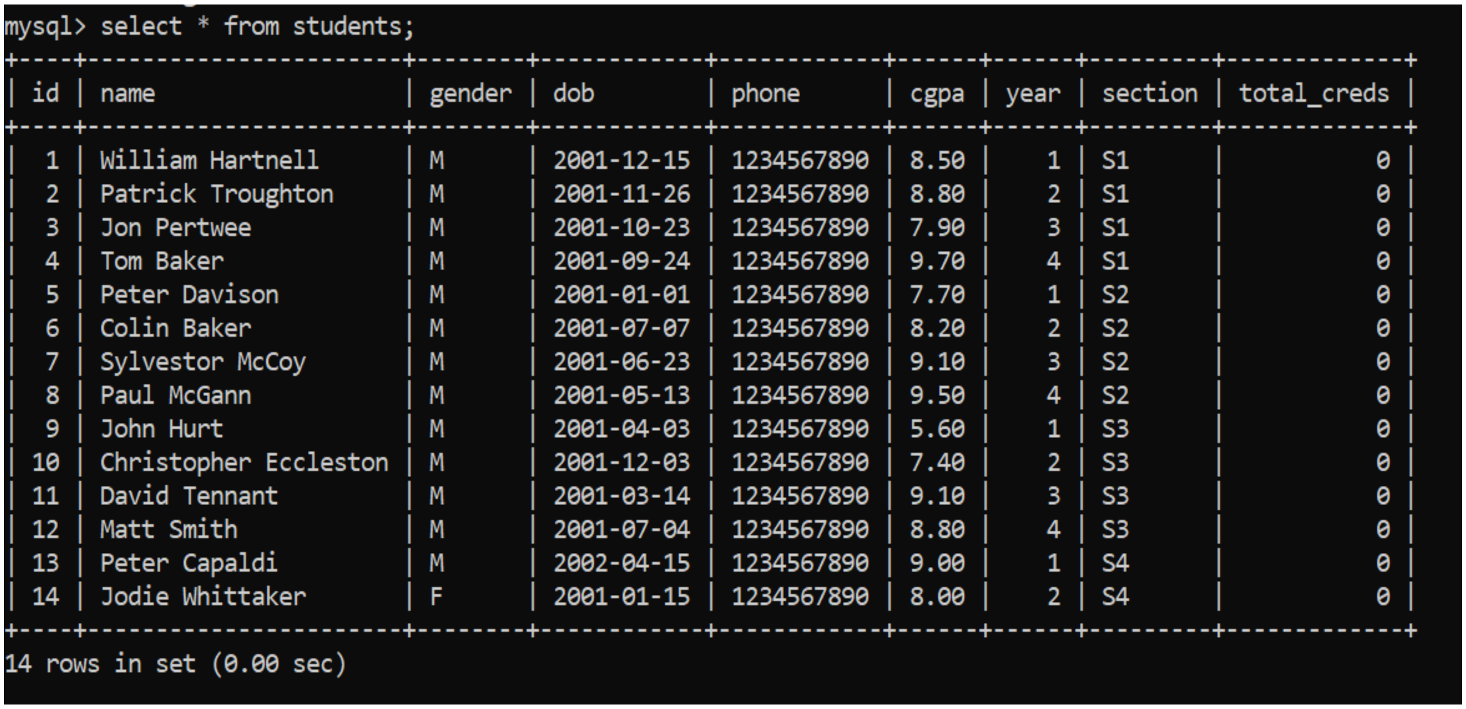
6. *Enrolled*
   Stores relations between the Student table and Courses table along with the gpa obtained in it. It stores the details of courses that a student is enrolled in. `student_id` is a foreign key to the Students table. `course_id` is a foreign key to the Courses table.



```
mysql> desc enrolled;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| id         | int          | NO   | PRI | NULL    | auto_increment |
| student_id | int          | NO   | MUL | NULL    |                |
| course_id  | int          | NO   | MUL | NULL    |                |
| gpa        | decimal(5,2) | YES  |     | 0.00    |                |
+------------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

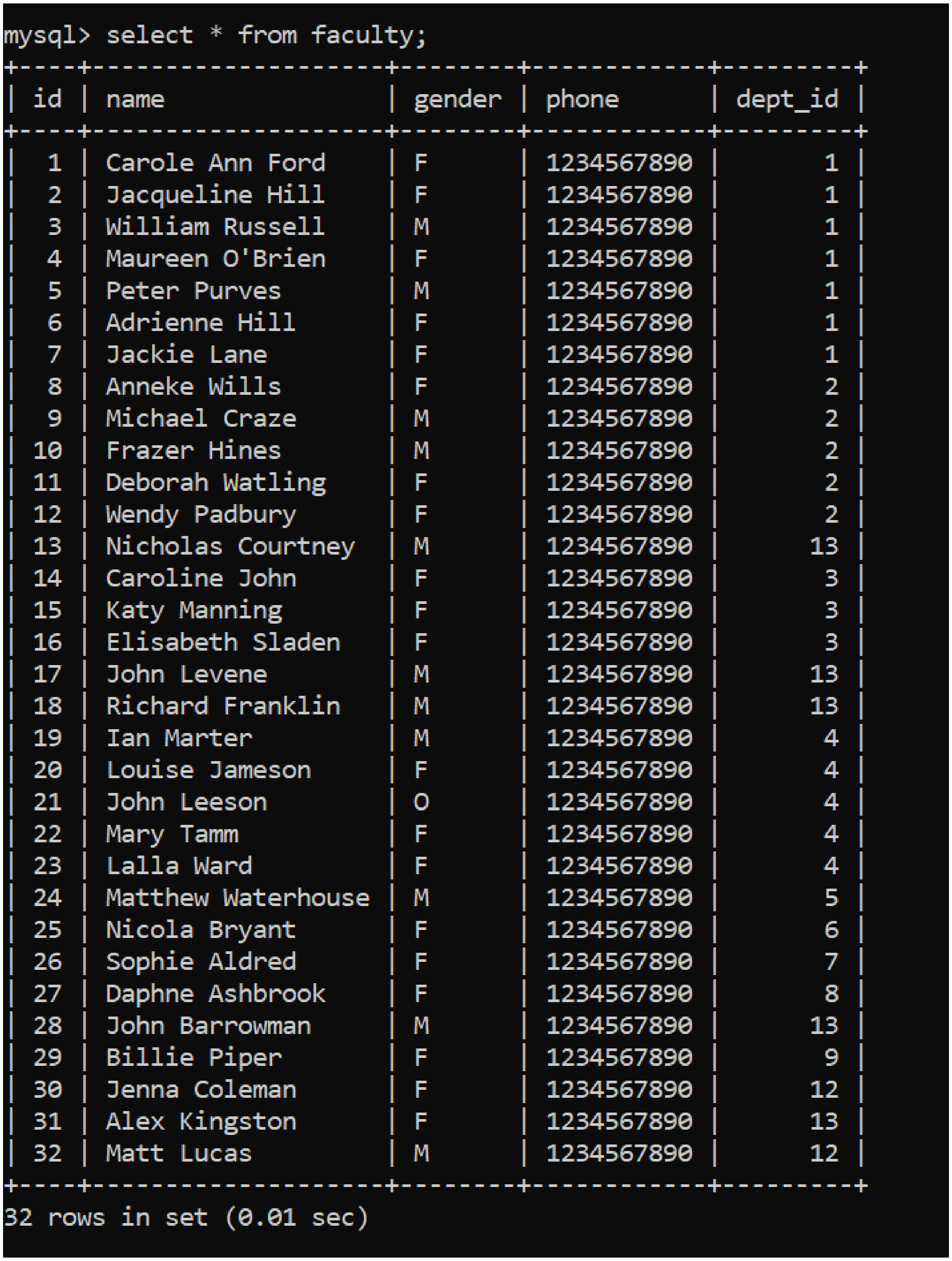
Table Screenshots

1. Students



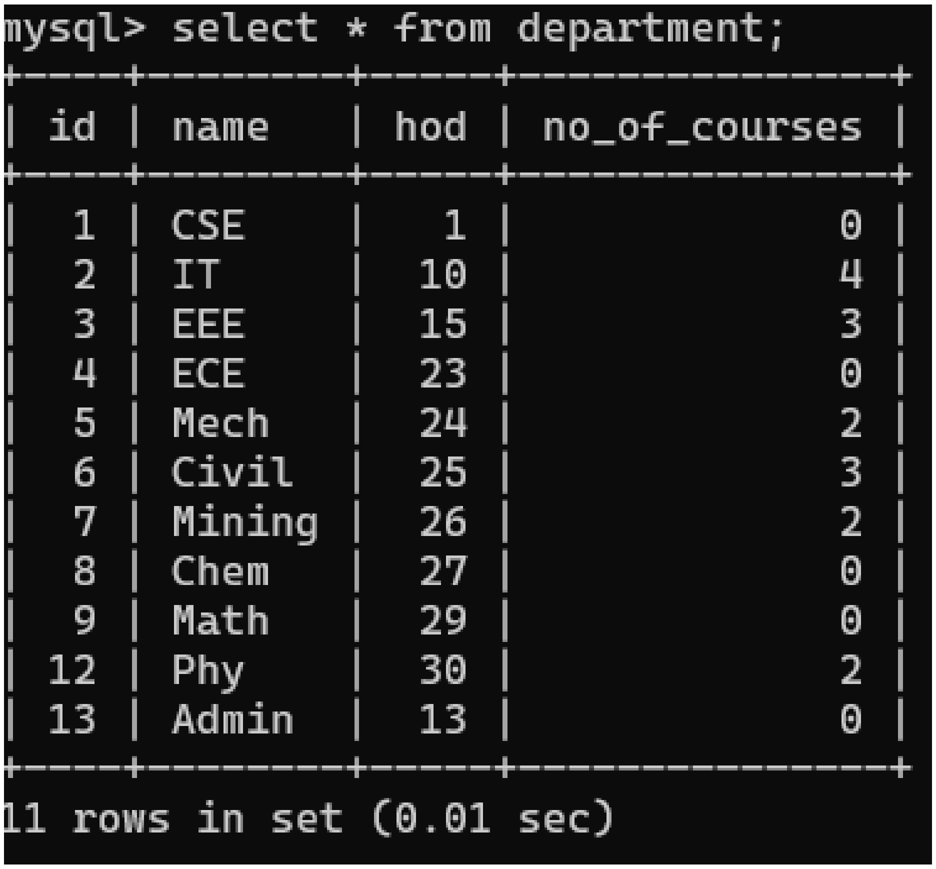
```
mysql> select * from students;
+----+---------------------+--------+------------+------------+------+------+---------+-------------+
| id | name                | gender | dob        | phone      | cgpa | year | section | total_creds |
+----+---------------------+--------+------------+------------+------+------+---------+-------------+
|  1 | William Hartnell    | M      | 2001-12-15 | 1234567890 | 8.50 |    1 | S1      |           0 |
|  2 | Patrick Troughton   | M      | 2001-11-26 | 1234567890 | 8.80 |    2 | S1      |           0 |
|  3 | Jon Pertwee         | M      | 2001-10-23 | 1234567890 | 7.90 |    3 | S1      |           0 |
|  4 | Tom Baker           | M      | 2001-09-24 | 1234567890 | 9.70 |    4 | S1      |           0 |
|  5 | Peter Davison       | M      | 2001-01-01 | 1234567890 | 7.70 |    1 | S2      |           0 |
|  6 | Colin Baker         | M      | 2001-07-07 | 1234567890 | 8.20 |    2 | S2      |           0 |
|  7 | Sylvestor McCoy     | M      | 2001-06-23 | 1234567890 | 9.10 |    3 | S2      |           0 |
|  8 | Paul McGann         | M      | 2001-05-13 | 1234567890 | 9.50 |    4 | S2      |           0 |
|  9 | John Hurt           | M      | 2001-04-03 | 1234567890 | 5.60 |    1 | S3      |           0 |
| 10 | Christopher Eccleston | M    | 2001-12-03 | 1234567890 | 7.40 |    2 | S3      |           0 |
| 11 | David Tennant       | M      | 2001-03-14 | 1234567890 | 9.10 |    3 | S3      |           0 |
| 12 | Matt Smith          | M      | 2001-07-04 | 1234567890 | 8.80 |    4 | S3      |           0 |
| 13 | Peter Capaldi       | M      | 2002-04-15 | 1234567890 | 9.00 |    1 | S4      |           0 |
| 14 | Jodie Whittaker     | F      | 2001-01-15 | 1234567890 | 8.00 |    2 | S4      |           0 |
+----+---------------------+--------+------------+------------+------+------+---------+-------------+
14 rows in set (0.00 sec)
```

2. Faculty



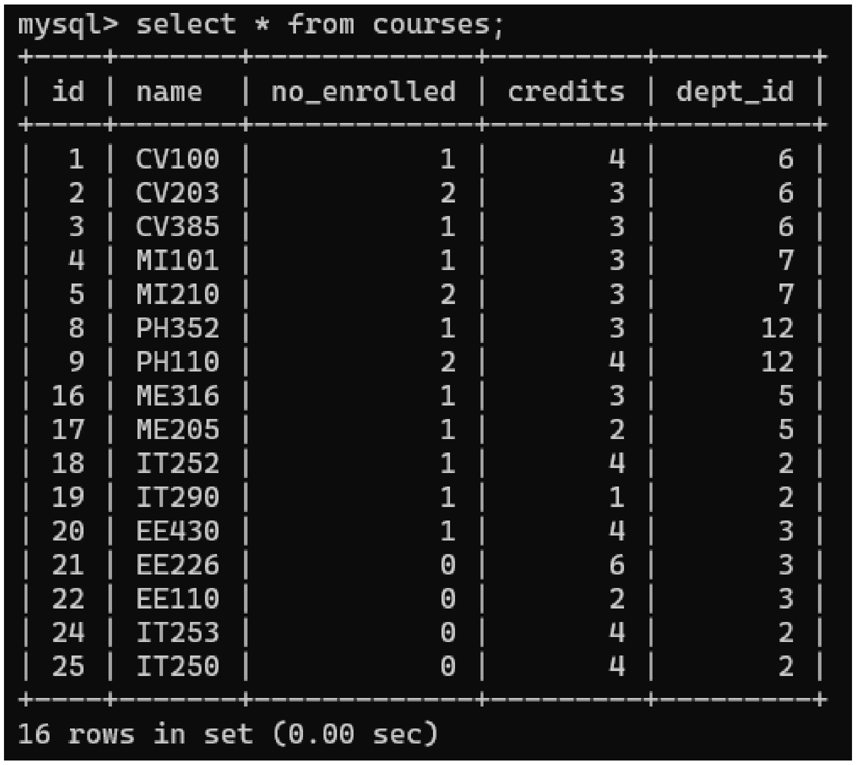
```
mysql> select * from faculty;
+----+--------------------+--------+------------+---------+
| id | name               | gender | phone      | dept_id |
+----+--------------------+--------+------------+---------+
|  1 | Carole Ann Ford    | F      | 1234567890 |       1 |
|  2 | Jacqueline Hill    | F      | 1234567890 |       1 |
|  3 | William Russell    | M      | 1234567890 |       1 |
|  4 | Maureen O'Brien    | F      | 1234567890 |       1 |
|  5 | Peter Purves       | M      | 1234567890 |       1 |
|  6 | Adrienne Hill      | F      | 1234567890 |       1 |
|  7 | Jackie Lane        | F      | 1234567890 |       1 |
|  8 | Anneke Wills       | F      | 1234567890 |       2 |
|  9 | Michael Craze      | M      | 1234567890 |       2 |
| 10 | Frazer Hines       | M      | 1234567890 |       2 |
| 11 | Deborah Watling    | F      | 1234567890 |       2 |
| 12 | Wendy Padbury      | F      | 1234567890 |       2 |
| 13 | Nicholas Courtney  | M      | 1234567890 |      13 |
| 14 | Caroline John      | F      | 1234567890 |       3 |
| 15 | Katy Manning       | F      | 1234567890 |       3 |
| 16 | Elisabeth Sladen   | F      | 1234567890 |       3 |
| 17 | John Levene        | M      | 1234567890 |      13 |
| 18 | Richard Franklin   | M      | 1234567890 |      13 |
| 19 | Ian Marter         | M      | 1234567890 |       4 |
| 20 | Louise Jameson     | F      | 1234567890 |       4 |
| 21 | John Leeson        | O      | 1234567890 |       4 |
| 22 | Mary Tamm          | F      | 1234567890 |       4 |
| 23 | Lalla Ward         | F      | 1234567890 |       4 |
| 24 | Matthew Waterhouse | M      | 1234567890 |       5 |
| 25 | Nicola Bryant      | F      | 1234567890 |       6 |
| 26 | Sophie Aldred      | F      | 1234567890 |       7 |
| 27 | Daphne Ashbrook    | F      | 1234567890 |       8 |
| 28 | John Barrowman     | M      | 1234567890 |      13 |
| 29 | Billie Piper       | F      | 1234567890 |       9 |
| 30 | Jenna Coleman      | F      | 1234567890 |      12 |
| 31 | Alex Kingston      | F      | 1234567890 |      13 |
| 32 | Matt Lucas         | M      | 1234567890 |      12 |
+----+--------------------+--------+------------+---------+
32 rows in set (0.01 sec)
```

3. Department



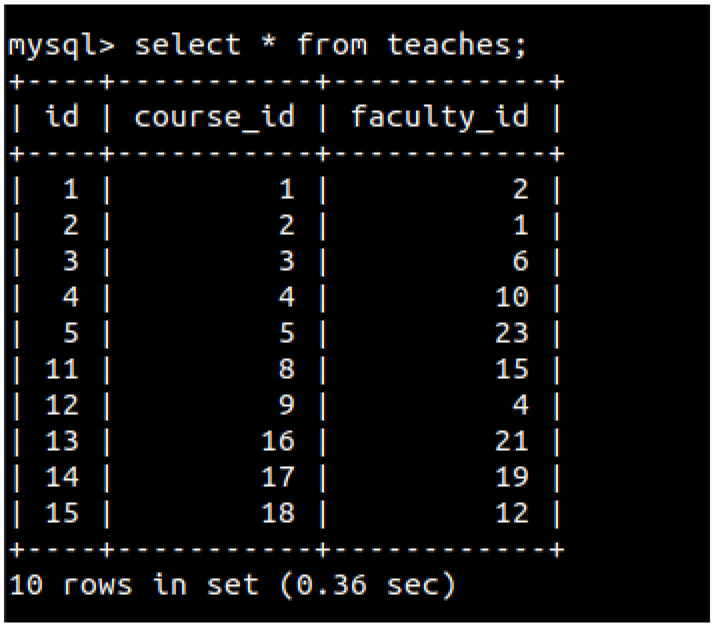
```
mysql> select * from department;
+----+--------+-----+---------------+
| id | name   | hod | no_of_courses |
+----+--------+-----+---------------+
|  1 | CSE    |   1 |             0 |
|  2 | IT     |  10 |             4 |
|  3 | EEE    |  15 |             3 |
|  4 | ECE    |  23 |             0 |
|  5 | Mech   |  24 |             2 |
|  6 | Civil  |  25 |             3 |
|  7 | Mining |  26 |             2 |
|  8 | Chem   |  27 |             0 |
|  9 | Math   |  29 |             0 |
| 12 | Phy    |  30 |             2 |
| 13 | Admin  |  13 |             0 |
+----+--------+-----+---------------+
11 rows in set (0.01 sec)
```

4. Courses



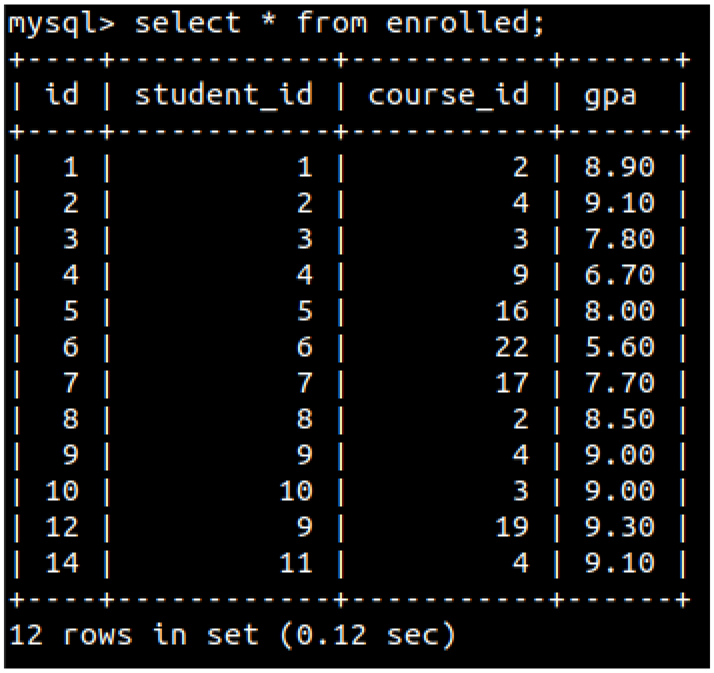
```
mysql> select * from courses;
+----+-------+-------------+---------+---------+
| id | name  | no_enrolled | credits | dept_id |
+----+-------+-------------+---------+---------+
|  1 | CV100 |           1 |       4 |       6 |
|  2 | CV203 |           2 |       3 |       6 |
|  3 | CV385 |           1 |       3 |       6 |
|  4 | MI101 |           1 |       3 |       7 |
|  5 | MI210 |           2 |       3 |       7 |
|  8 | PH352 |           1 |       3 |      12 |
|  9 | PH110 |           2 |       4 |      12 |
| 16 | ME316 |           1 |       3 |       5 |
| 17 | ME205 |           1 |       2 |       5 |
| 18 | IT252 |           1 |       4 |       2 |
| 19 | IT290 |           1 |       1 |       2 |
| 20 | EE430 |           1 |       4 |       3 |
| 21 | EE226 |           0 |       6 |       3 |
| 22 | EE110 |           0 |       2 |       3 |
| 24 | IT253 |           0 |       4 |       2 |
| 25 | IT250 |           0 |       4 |       2 |
+----+-------+-------------+---------+---------+
16 rows in set (0.00 sec)
```

5. Teaches



```
mysql> select * from teaches;
+----+-----------+------------+
| id | course_id | faculty_id |
+----+-----------+------------+
|  1 |         1 |          2 |
|  2 |         2 |          1 |
|  3 |         3 |          6 |
|  4 |         4 |         10 |
|  5 |         5 |         23 |
| 11 |         8 |         15 |
| 12 |         9 |          4 |
| 13 |        16 |         21 |
| 14 |        17 |         19 |
| 15 |        18 |         12 |
+----+-----------+------------+
10 rows in set (0.36 sec)
```

6. Enrolled

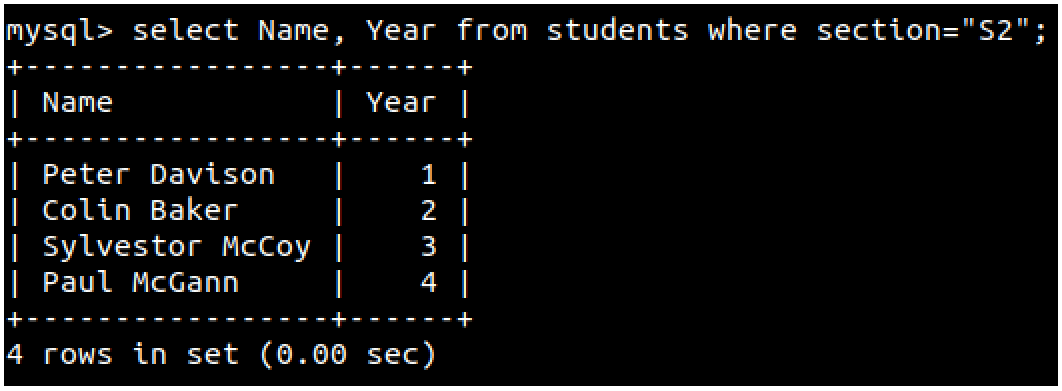


```
mysql> select * from enrolled;
+----+------------+-----------+------+
| id | student_id | course_id | gpa  |
+----+------------+-----------+------+
|  1 |          1 |         2 | 8.90 |
|  2 |          2 |         4 | 9.10 |
|  3 |          3 |         3 | 7.80 |
|  4 |          4 |         9 | 6.70 |
|  5 |          5 |        16 | 8.00 |
|  6 |          6 |        22 | 5.60 |
|  7 |          7 |        17 | 7.70 |
|  8 |          8 |         2 | 8.50 |
|  9 |          9 |         4 | 9.00 |
| 10 |         10 |         3 | 9.00 |
| 12 |          9 |        19 | 9.30 |
| 14 |         11 |         4 | 9.10 |
+----+------------+-----------+------+
12 rows in set (0.12 sec)
```

Simple Queries

1. List out all students along with their names and years, belonging to Section S2

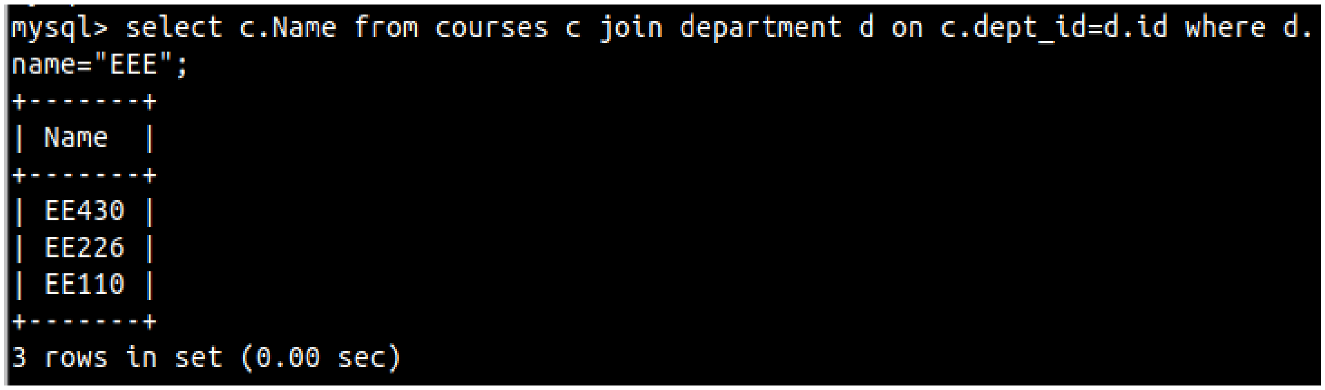
A. `select Name, Year from students where section="S2";`



```
mysql> select Name, Year from students where section="S2";
+-----------------+------+
| Name            | Year |
+-----------------+------+
| Peter Davison   |    1 |
| Colin Baker     |    2 |
| Sylvestor McCoy |    3 |
| Paul McGann     |    4 |
+-----------------+------+
4 rows in set (0.00 sec)
```

2. Display all courses belonging to the EEE department

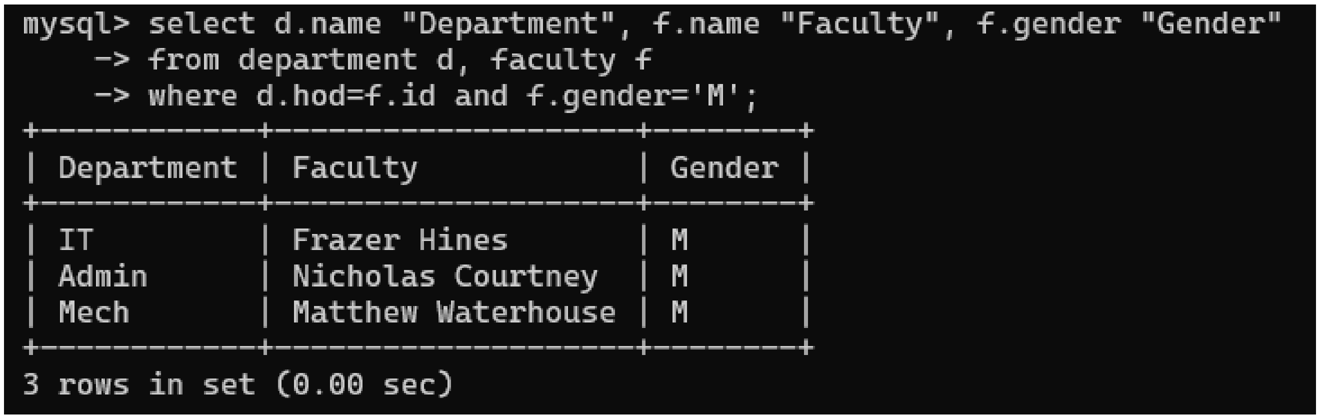
A. `select c.Name from courses c join department d on c.dept_id=d.id where d.name="EEE";`



```
mysql> select c.Name from courses c join department d on c.dept_id=d.id where d.
name="EEE";
+-------+
| Name  |
+-------+
| EE430 |
| EE226 |
| EE110 |
+-------+
3 rows in set (0.00 sec)
```

3. Display the name of male HODs and their respective department

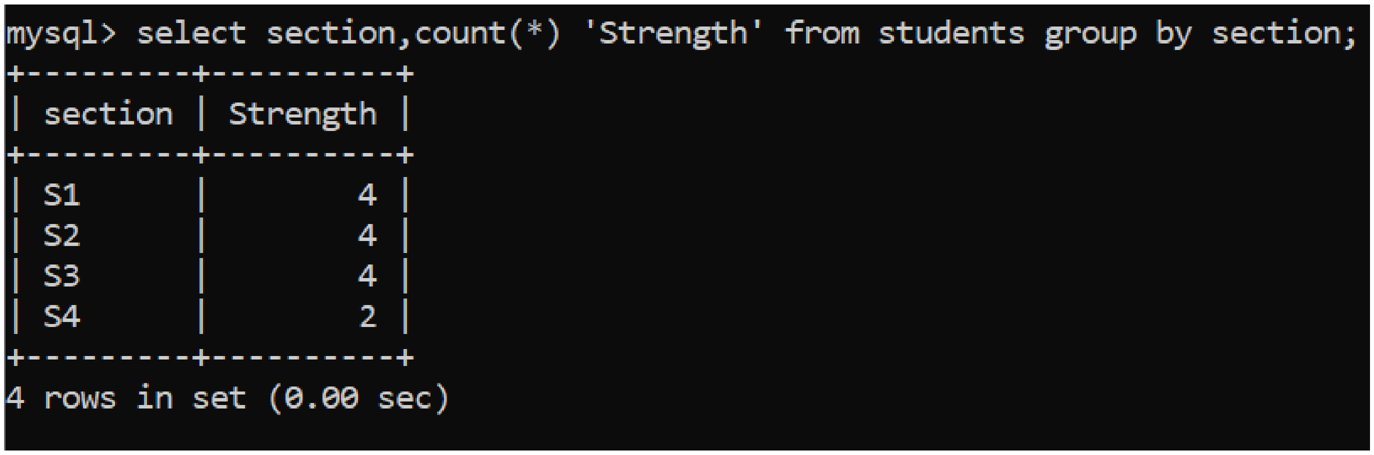
A. `select d.name "Department", f.name "Faculty", f.gender "Gender" from department d, faculty f where d.hod=f.id and f.gender='M';`



```
mysql> select d.name "Department", f.name "Faculty", f.gender "Gender"
    -> from department d, faculty f
    -> where d.hod=f.id and f.gender='M';
+------------+---------------------+--------+
| Department | Faculty             | Gender |
+------------+---------------------+--------+
| IT         | Frazer Hines        | M      |
| Admin      | Nicholas Courtney   | M      |
| Mech       | Matthew Waterhouse  | M      |
+------------+---------------------+--------+
3 rows in set (0.00 sec)
```

4. Display the strength of each section.

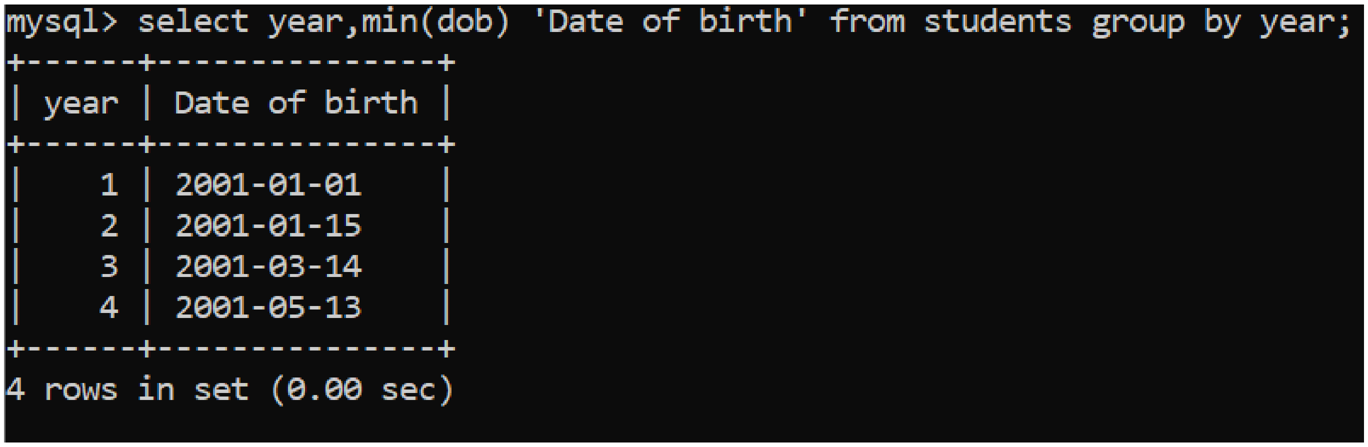
A. `select section,count(*) 'Strength' from students group by section;`



```
mysql> select section,count(*) 'Strength' from students group by section;
+---------+----------+
| section | Strength |
+---------+----------+
| S1      |        4 |
| S2      |        4 |
| S3      |        4 |
| S4      |        2 |
+---------+----------+
4 rows in set (0.00 sec)
```

5. Display the DOB of the oldest student in each year.

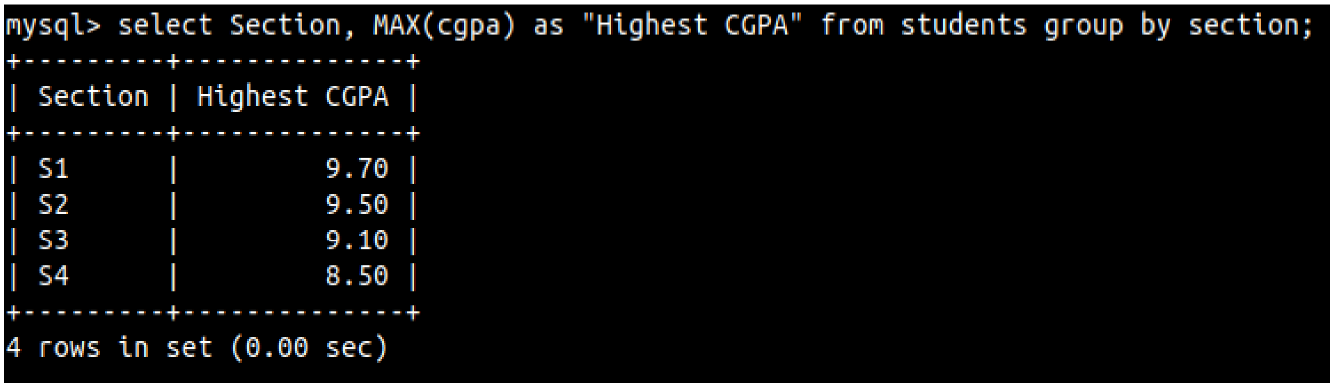
A. `select year,min(dob) 'Date of birth' from students group by year;`



```
mysql> select year,min(dob) 'Date of birth' from students group by year;
+------+---------------+
| year | Date of birth |
+------+---------------+
|    1 | 2001-01-01    |
|    2 | 2001-01-15    |
|    3 | 2001-03-14    |
|    4 | 2001-05-13    |
+------+---------------+
4 rows in set (0.00 sec)
```

6. Display highest CGPAs from each section

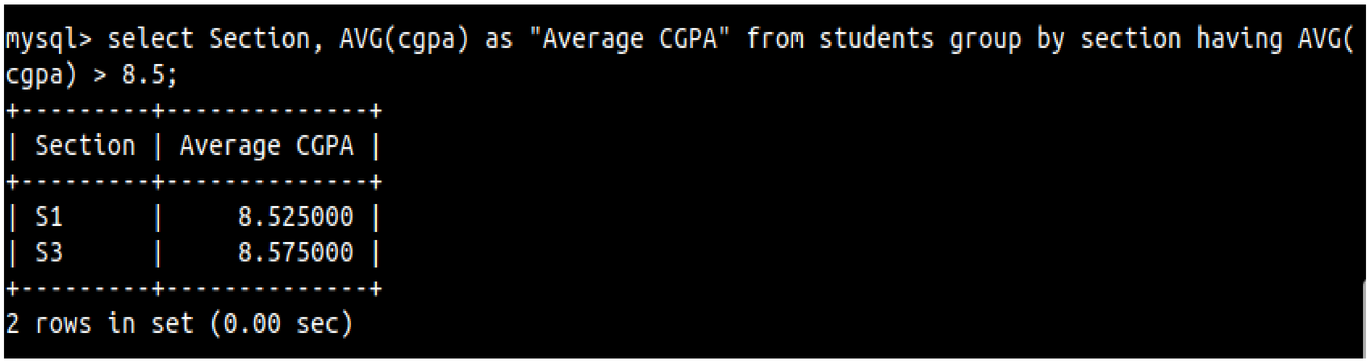
A. `select Section, MAX(cgpa) as "Highest CGPA" from students group by section;`



```
mysql> select Section, MAX(cgpa) as "Highest CGPA" from students group by section;
+---------+--------------+
| Section | Highest CGPA |
+---------+--------------+
| S1      |         9.70 |
| S2      |         9.50 |
| S3      |         9.10 |
| S4      |         8.50 |
+---------+--------------+
4 rows in set (0.00 sec)
```

7. Display sections having average CGPA greater than 8.5

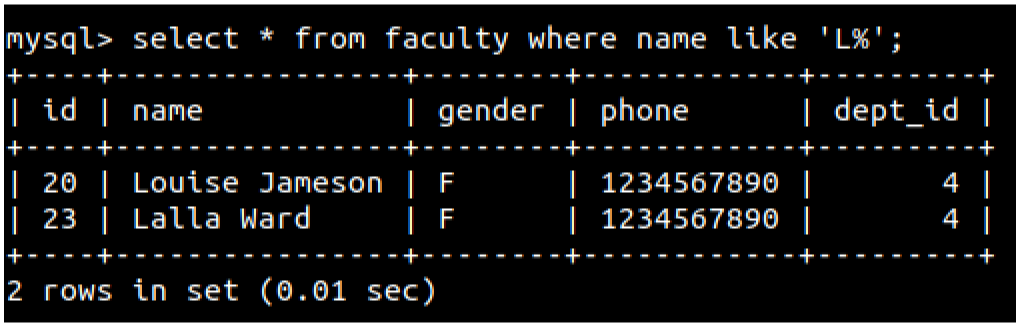
A. `select Section, AVG(cgpa) as "Average CGPA" from students group by section having AVG(cgpa) > 8.5;`



```
mysql> select Section, AVG(cgpa) as "Average CGPA" from students group by section having AVG(
cgpa) > 8.5;
+---------+--------------+
| Section | Average CGPA |
+---------+--------------+
| S1      |     8.525000 |
| S3      |     8.575000 |
+---------+--------------+
2 rows in set (0.00 sec)
```

8. Display all details for faculty whose name starts with L

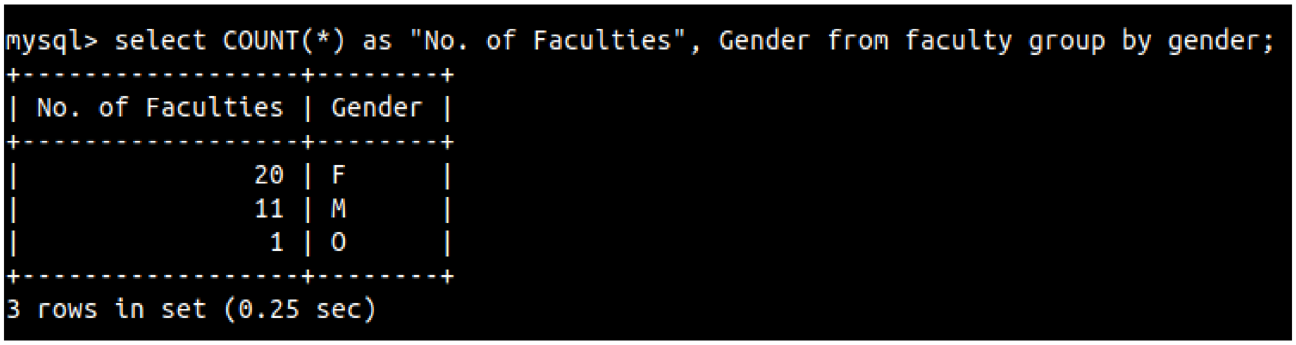
A. `select * from faculty where name like 'L%';`



```
mysql> select * from faculty where name like 'L%';
+----+-----------------+--------+------------+---------+
| id | name            | gender | phone      | dept_id |
+----+-----------------+--------+------------+---------+
| 20 | Louise Jameson  | F      | 1234567890 |       4 |
| 23 | Lalla Ward      | F      | 1234567890 |       4 |
+----+-----------------+--------+------------+---------+
2 rows in set (0.01 sec)
```

9. Display number of faculties of each gender

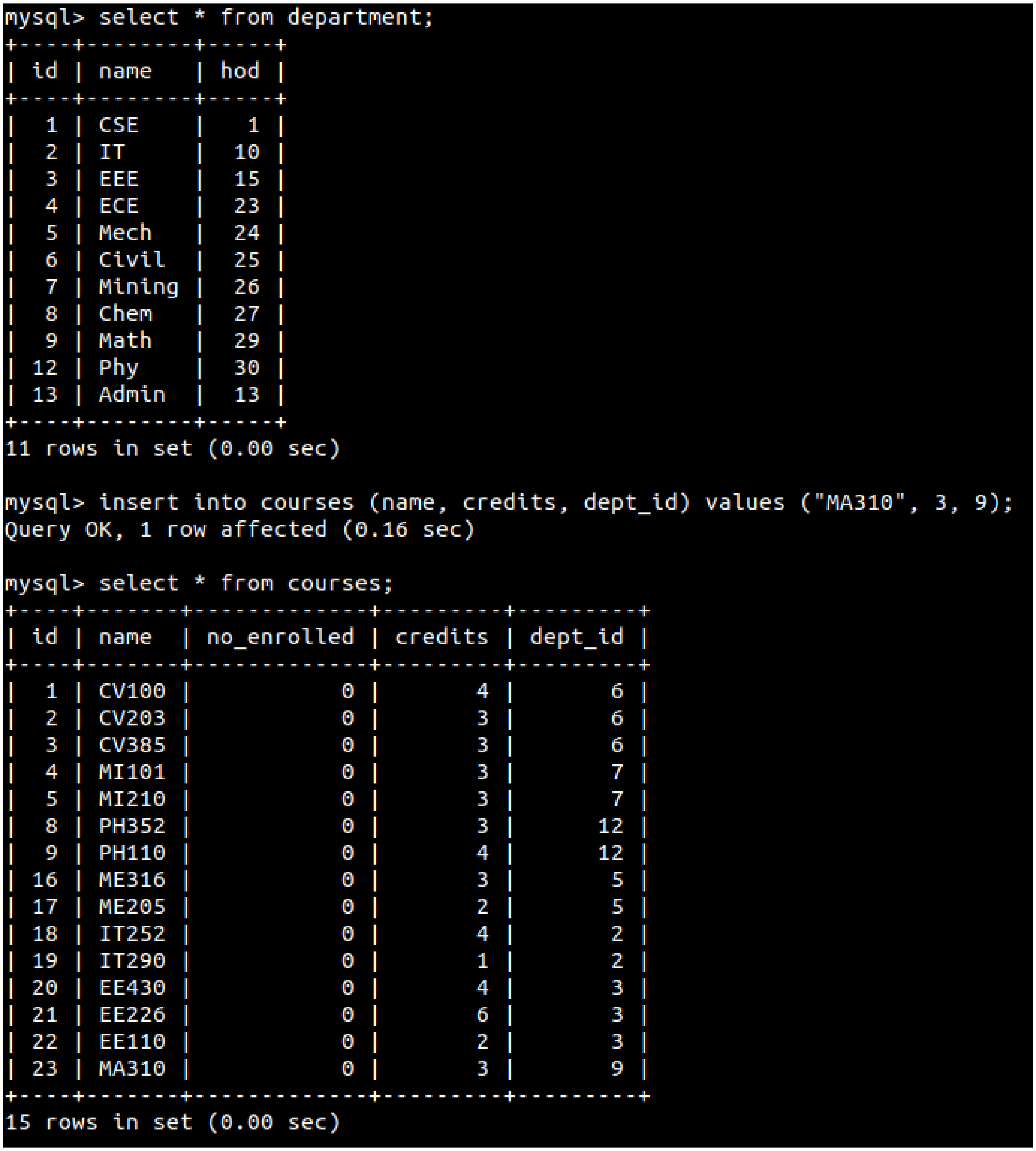
A. `select COUNT(*) as "No. of Faculties", Gender from faculty group by gender;`



```
mysql> select COUNT(*) as "No. of Faculties", Gender from faculty group by gender;
+------------------+--------+
| No. of Faculties | Gender |
+------------------+--------+
|               20 | F      |
|               11 | M      |
|                1 | O      |
+------------------+--------+
3 rows in set (0.25 sec)
```

10. Add a new course "MA310" belonging to the Math Department

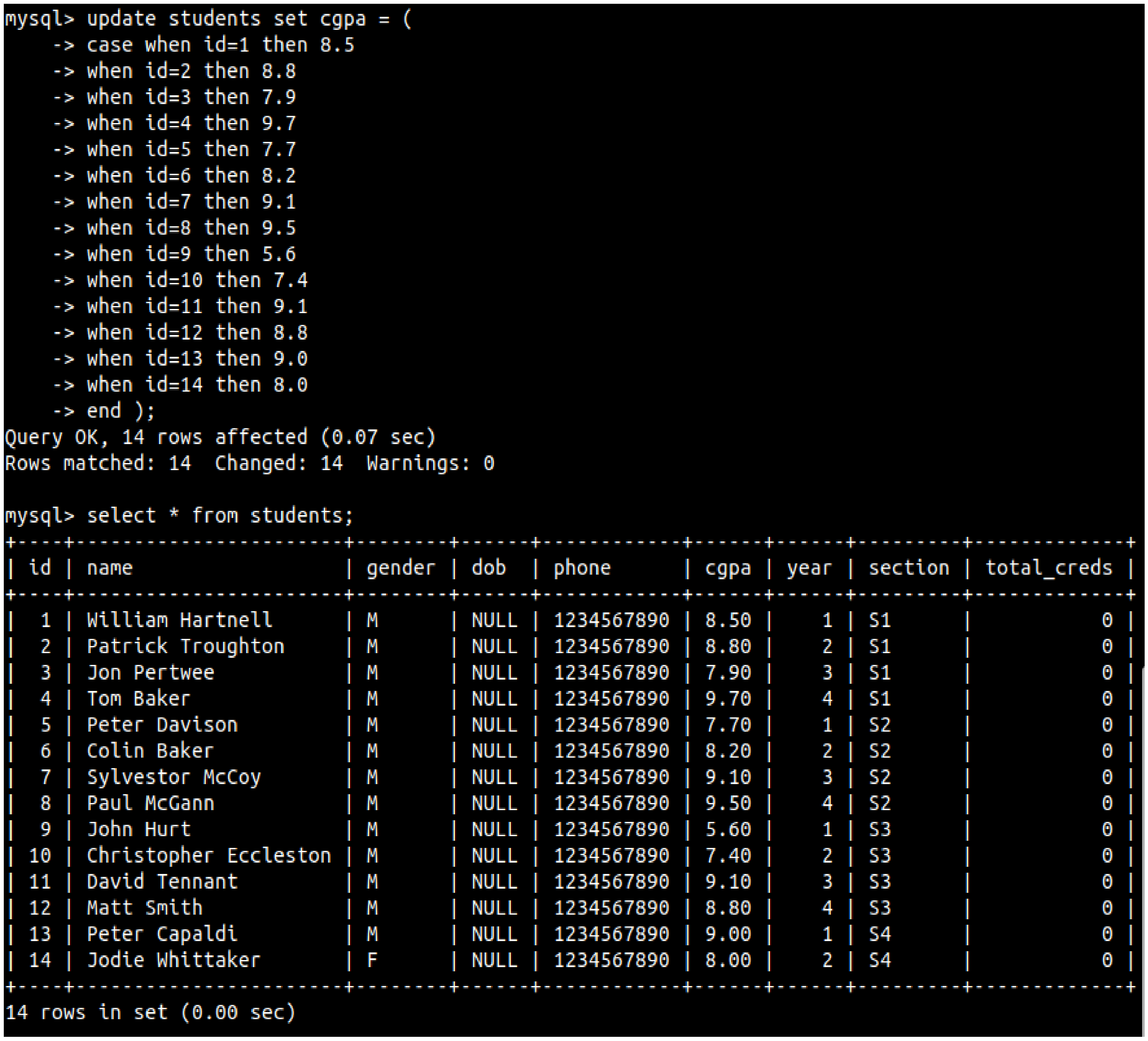
A. `insert into courses (name, credits, dept_id) values ("MA310", 3, 9);`



```
mysql> select * from department;
+----+--------+-----+
| id | name   | hod |
+----+--------+-----+
|  1 | CSE    |   1 |
|  2 | IT     |  10 |
|  3 | EEE    |  15 |
|  4 | ECE    |  23 |
|  5 | Mech   |  24 |
|  6 | Civil  |  25 |
|  7 | Mining |  26 |
|  8 | Chem   |  27 |
|  9 | Math   |  29 |
| 12 | Phy    |  30 |
| 13 | Admin  |  13 |
+----+--------+-----+
11 rows in set (0.00 sec)

mysql> insert into courses (name, credits, dept_id) values ("MA310", 3, 9);
Query OK, 1 row affected (0.16 sec)

mysql> select * from courses;
+----+-------+-------------+---------+---------+
| id | name  | no_enrolled | credits | dept_id |
+----+-------+-------------+---------+---------+
|  1 | CV100 |           0 |       4 |       6 |
|  2 | CV203 |           0 |       3 |       6 |
|  3 | CV385 |           0 |       3 |       6 |
|  4 | MI101 |           0 |       3 |       7 |
|  5 | MI210 |           0 |       3 |       7 |
|  8 | PH352 |           0 |       3 |      12 |
|  9 | PH110 |           0 |       4 |      12 |
| 16 | ME316 |           0 |       3 |       5 |
| 17 | ME205 |           0 |       2 |       5 |
| 18 | IT252 |           0 |       4 |       2 |
| 19 | IT290 |           0 |       1 |       2 |
| 20 | EE430 |           0 |       4 |       3 |
| 21 | EE226 |           0 |       6 |       3 |
| 22 | EE110 |           0 |       2 |       3 |
| 23 | MA310 |           0 |       3 |       9 |
+----+-------+-------------+---------+---------+
15 rows in set (0.00 sec)
```

11. Update CGPA of all students

A.
```
update students set cgpa = (
case when id=1 then 8.5
     when id=2 then 8.8
     when id=3 then 7.9
     when id=4 then 9.7
     when id=5 then 7.7
     when id=6 then 8.2
     when id=7 then 9.1
     when id=8 then 9.5
     when id=9 then 5.6
     when id=10 then 7.4
     when id=11 then 9.1
     when id=12 then 8.8
     when id=13 then 9.0
     when id=14 then 8.0
end );
```



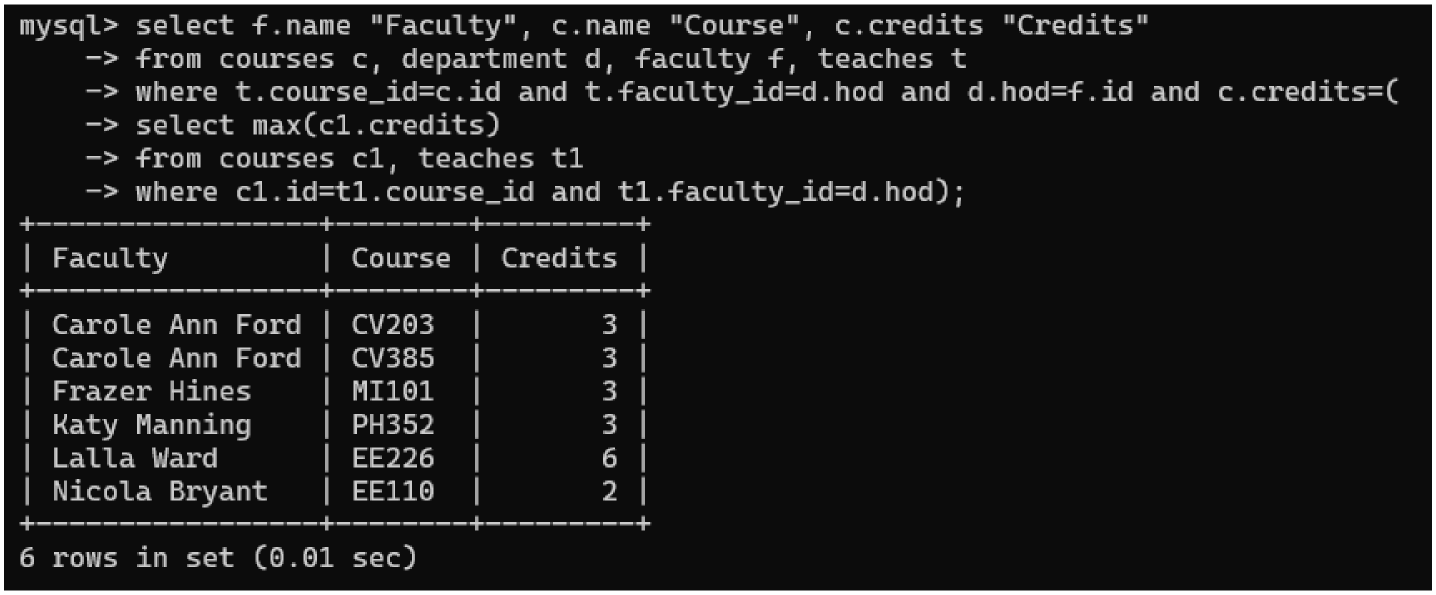
```
mysql> update students set cgpa = (
    -> case when id=1 then 8.5
    -> when id=2 then 8.8
    -> when id=3 then 7.9
    -> when id=4 then 9.7
    -> when id=5 then 7.7
    -> when id=6 then 8.2
    -> when id=7 then 9.1
    -> when id=8 then 9.5
    -> when id=9 then 5.6
    -> when id=10 then 7.4
    -> when id=11 then 9.1
    -> when id=12 then 8.8
    -> when id=13 then 9.0
    -> when id=14 then 8.0
    -> end );
Query OK, 14 rows affected (0.07 sec)
Rows matched: 14  Changed: 14  Warnings: 0

mysql> select * from students;
+----+----------------------+--------+------+------------+------+------+---------+-------------+
| id | name                 | gender | dob  | phone      | cgpa | year | section | total_creds |
+----+----------------------+--------+------+------------+------+------+---------+-------------+
|  1 | William Hartnell     | M      | NULL | 1234567890 | 8.50 |    1 | S1      |           0 |
|  2 | Patrick Troughton    | M      | NULL | 1234567890 | 8.80 |    2 | S1      |           0 |
|  3 | Jon Pertwee          | M      | NULL | 1234567890 | 7.90 |    3 | S1      |           0 |
|  4 | Tom Baker            | M      | NULL | 1234567890 | 9.70 |    4 | S1      |           0 |
|  5 | Peter Davison        | M      | NULL | 1234567890 | 7.70 |    1 | S2      |           0 |
|  6 | Colin Baker          | M      | NULL | 1234567890 | 8.20 |    2 | S2      |           0 |
|  7 | Sylvestor McCoy      | M      | NULL | 1234567890 | 9.10 |    3 | S2      |           0 |
|  8 | Paul McGann          | M      | NULL | 1234567890 | 9.50 |    4 | S2      |           0 |
|  9 | John Hurt            | M      | NULL | 1234567890 | 5.60 |    1 | S3      |           0 |
| 10 | Christopher Eccleston | M     | NULL | 1234567890 | 7.40 |    2 | S3      |           0 |
| 11 | David Tennant        | M      | NULL | 1234567890 | 9.10 |    3 | S3      |           0 |
| 12 | Matt Smith           | M      | NULL | 1234567890 | 8.80 |    4 | S3      |           0 |
| 13 | Peter Capaldi        | M      | NULL | 1234567890 | 9.00 |    1 | S4      |           0 |
| 14 | Jodie Whittaker      | F      | NULL | 1234567890 | 8.00 |    2 | S4      |           0 |
+----+----------------------+--------+------+------------+------+------+---------+-------------+
14 rows in set (0.00 sec)
```

Complex Queries

Correlated subqueries

1. Courses taught by HODs having the highest credits

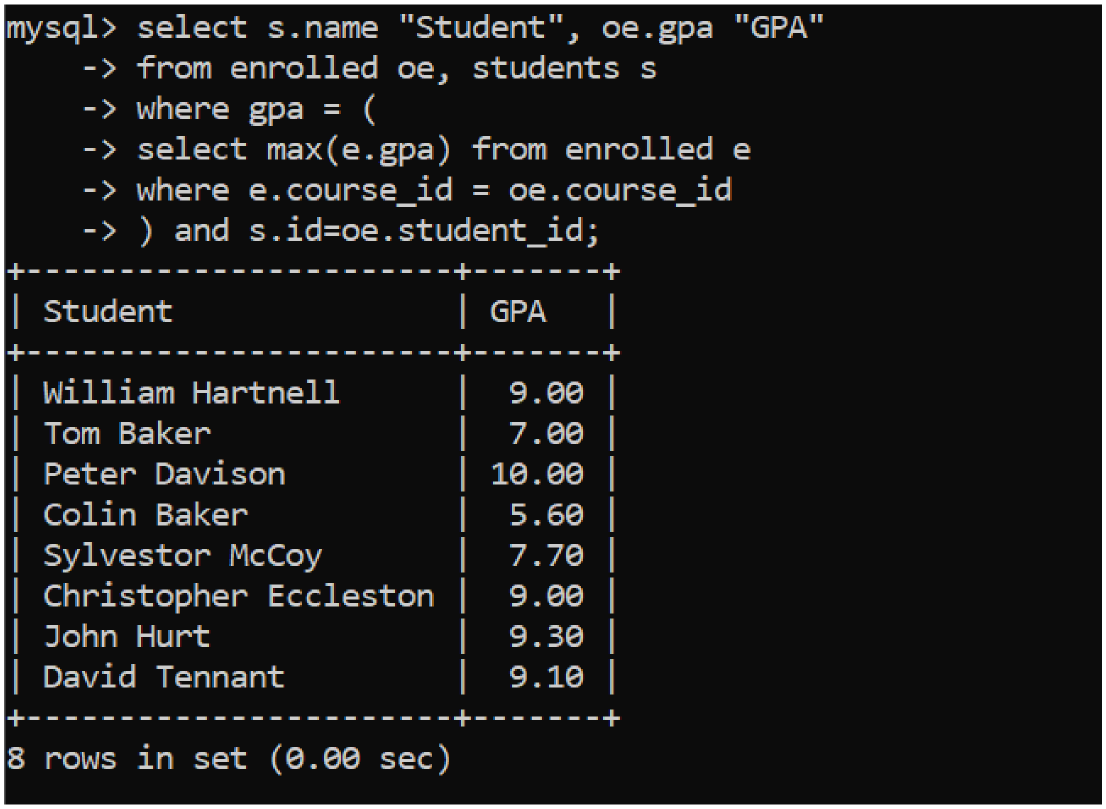
```
A. select f.name "Faculty", c.name "Course", c.credits "Credits"
   from courses c, department d, faculty f, teaches t
   where t.course_id=c.id and t.faculty_id=d.hod and d.hod=f.id and
   c.credits=(
   select max(c1.credits)
   from courses c1, teaches t1
   where c1.id=t1.course_id and t1.faculty_id=d.hod);
```



```
mysql> select f.name "Faculty", c.name "Course", c.credits "Credits"
    -> from courses c, department d, faculty f, teaches t
    -> where t.course_id=c.id and t.faculty_id=d.hod and d.hod=f.id and c.credits=(
    -> select max(c1.credits)
    -> from courses c1, teaches t1
    -> where c1.id=t1.course_id and t1.faculty_id=d.hod);
+-----------------+--------+---------+
| Faculty         | Course | Credits |
+-----------------+--------+---------+
| Carole Ann Ford | CV203  |       3 |
| Carole Ann Ford | CV385  |       3 |
| Frazer Hines    | MI101  |       3 |
| Katy Manning    | PH352  |       3 |
| Lalla Ward      | EE226  |       6 |
| Nicola Bryant   | EE110  |       2 |
+-----------------+--------+---------+
6 rows in set (0.01 sec)
```

2. Display name of student who has secured the highest GPA from each Course

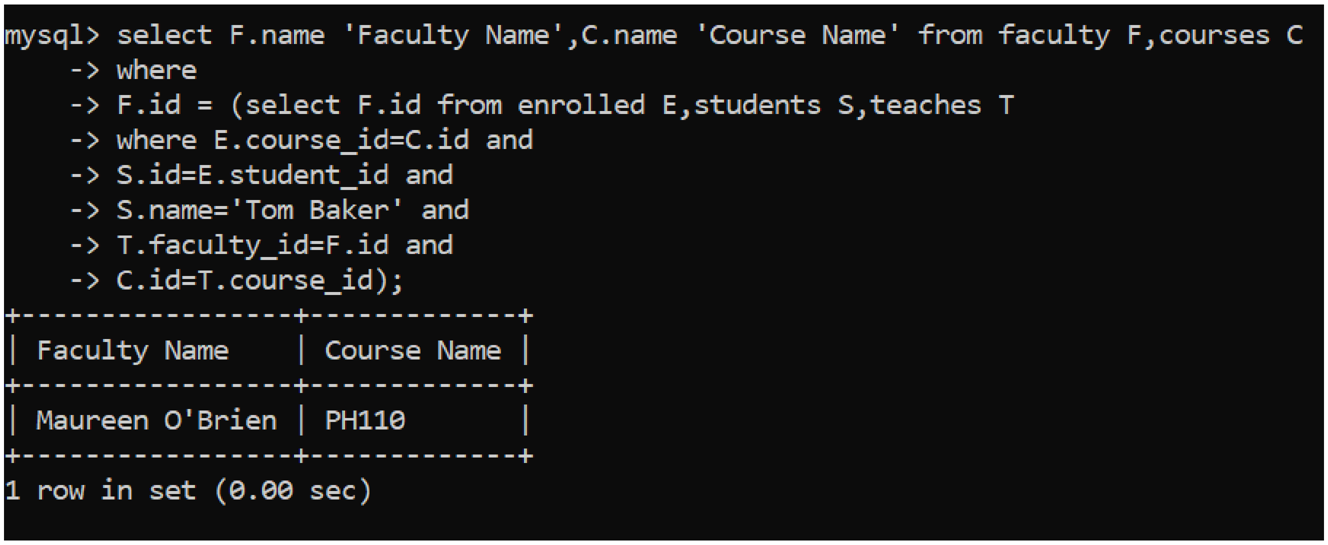
```
A. select s.name "Student", oe.gpa "GPA"
   from enrolled oe, students s
   where gpa = (
   select max(e.gpa) from enrolled e
   where e.course_id = oe.course_id
   ) and s.id=oe.student_id;
```



```
mysql> select s.name "Student", oe.gpa "GPA"
    -> from enrolled oe, students s
    -> where gpa = (
    -> select max(e.gpa) from enrolled e
    -> where e.course_id = oe.course_id
    -> ) and s.id=oe.student_id;
+-----------------------+-------+
| Student               | GPA   |
+-----------------------+-------+
| William Hartnell      |  9.00 |
| Tom Baker             |  7.00 |
| Peter Davison         | 10.00 |
| Colin Baker           |  5.60 |
| Sylvestor McCoy       |  7.70 |
| Christopher Eccleston |  9.00 |
| John Hurt             |  9.30 |
| David Tennant         |  9.10 |
+-----------------------+-------+
8 rows in set (0.00 sec)
```

3. Display the course name and faculty that teaches the course of the student named “Tom Baker”.

```
A. select F.name 'Faculty Name',C.name 'Course Name'
   from faculty F, courses C where
   F.id = (select F.id from enrolled E,students S,teaches T
   where E.course_id=C.id and
   S.id=E.student_id and
   S.name='Tom Baker' and
   T.faculty_id=F.id and
   C.id=T.course_id);
```

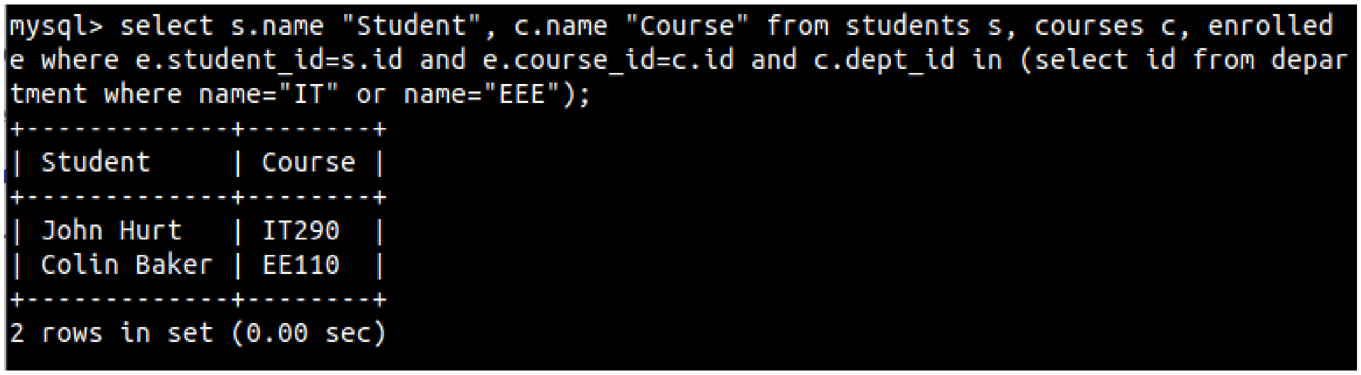


```
mysql> select F.name 'Faculty Name',C.name 'Course Name' from faculty F,courses C
    -> where
    -> F.id = (select F.id from enrolled E,students S,teaches T
    -> where E.course_id=C.id and
    -> S.id=E.student_id and
    -> S.name='Tom Baker' and
    -> T.faculty_id=F.id and
    -> C.id=T.course_id);
+-----------------+-------------+
| Faculty Name    | Course Name |
+-----------------+-------------+
| Maureen O'Brien | PH110       |
+-----------------+-------------+
1 row in set (0.00 sec)
```

Nested subqueries

4. Display names of students who have enrolled in IT and EEE courses

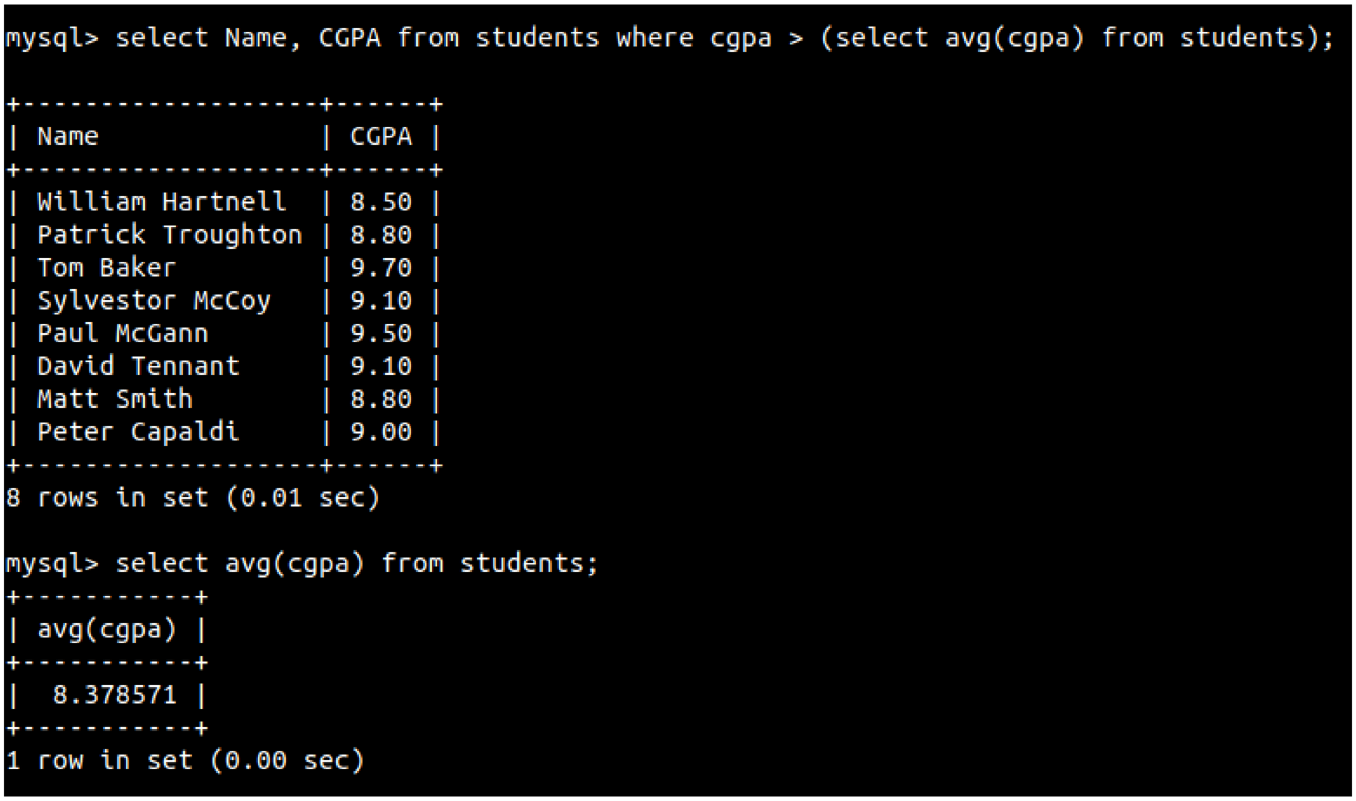
A. `select s.name "Student", c.name "Course" from students s, courses c, enrolled e where e.student_id=s.id and e.course_id=c.id and c.dept_id in (select id from department where name="IT" or name="EEE");`



```
mysql> select s.name "Student", c.name "Course" from students s, courses c, enrolled
e where e.student_id=s.id and e.course_id=c.id and c.dept_id in (select id from depar
tment where name="IT" or name="EEE");
+-------------+--------+
| Student     | Course |
+-------------+--------+
| John Hurt   | IT290  |
| Colin Baker | EE110  |
+-------------+--------+
2 rows in set (0.00 sec)
```

5. Display names of students who have a CGPA higher than average

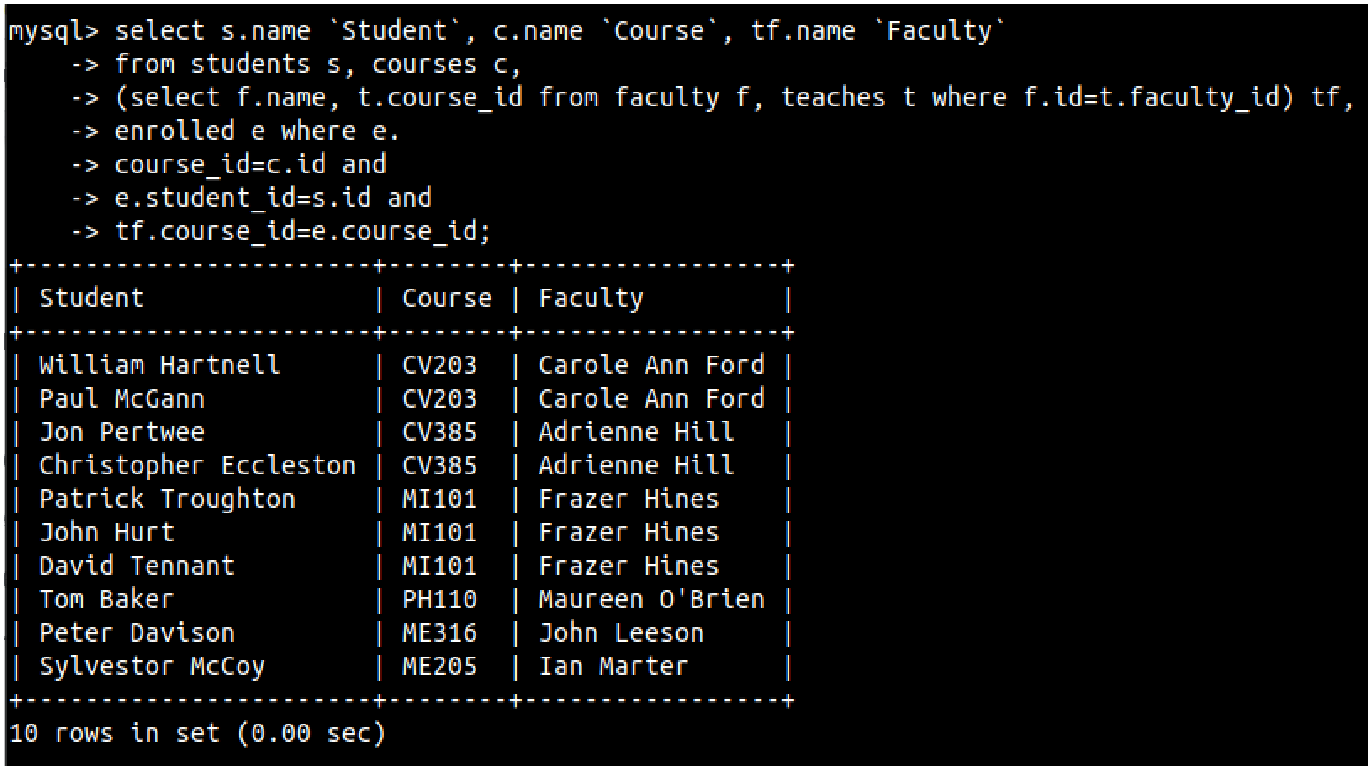
A. `select Name, CGPA from students where cgpa > (select avg(cgpa) from students);`



```
mysql> select Name, CGPA from students where cgpa > (select avg(cgpa) from students);

+-------------------+------+
| Name              | CGPA |
+-------------------+------+
| William Hartnell  | 8.50 |
| Patrick Troughton | 8.80 |
| Tom Baker         | 9.70 |
| Sylvestor McCoy   | 9.10 |
| Paul McGann       | 9.50 |
| David Tennant     | 9.10 |
| Matt Smith        | 8.80 |
| Peter Capaldi     | 9.00 |
+-------------------+------+
8 rows in set (0.01 sec)

mysql> select avg(cgpa) from students;
+-----------+
| avg(cgpa) |
+-----------+
|  8.378571 |
+-----------+
1 row in set (0.00 sec)
```

6. Display the names of students enrolled in a course along with course name and name of the faculty teaching the course

A.
```
select s.name `Student`, c.name `Course`, tf.name `Faculty`
from students s, courses c,
(select f.name, t.course_id from faculty f, teaches t where
f.id=t.faculty_id) tf,
enrolled e where e.
course_id=c.id and
e.student_id=s.id and
tf.course_id=e.course_id;
```

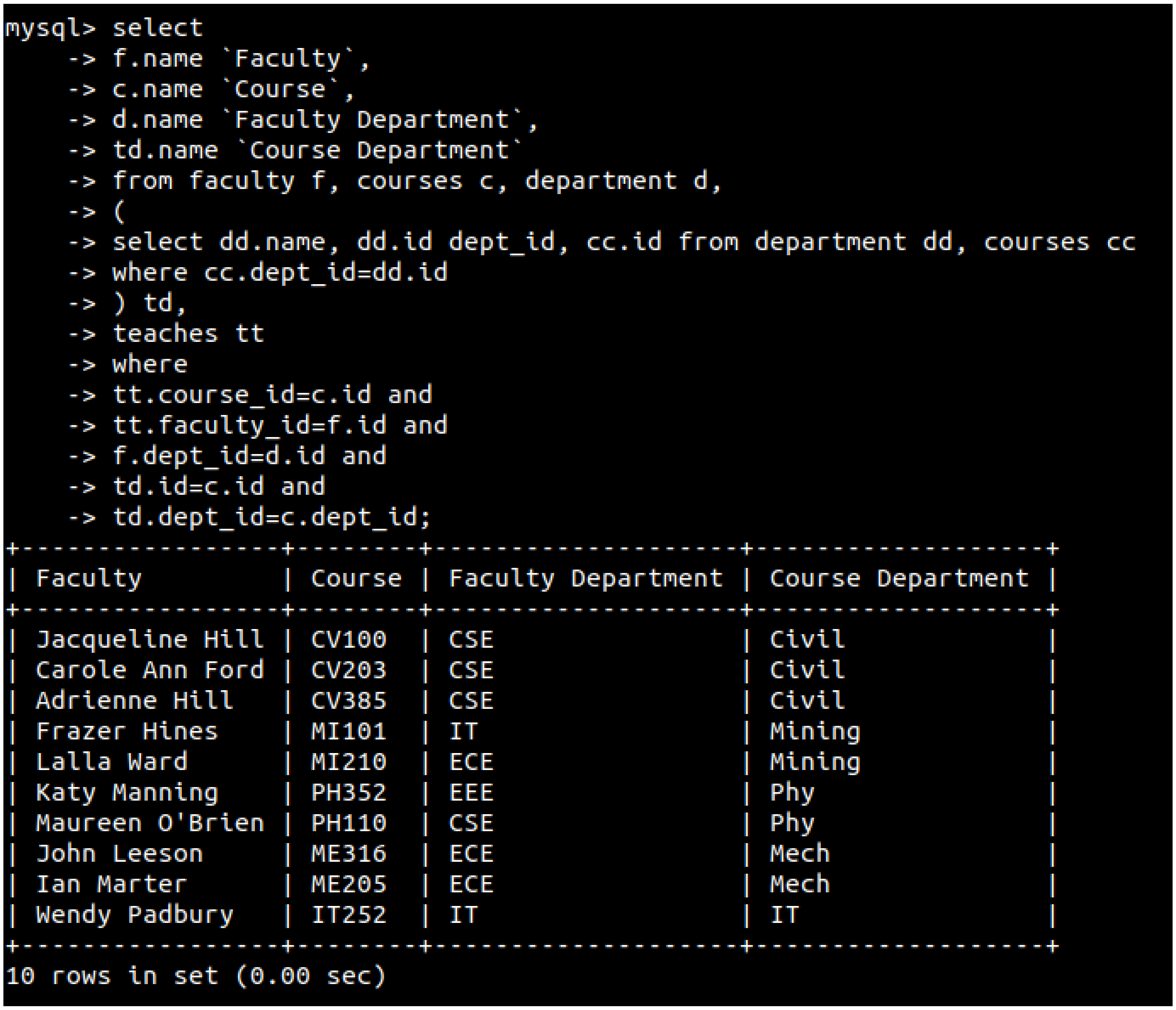


```
mysql> select s.name `Student`, c.name `Course`, tf.name `Faculty`
    -> from students s, courses c,
    -> (select f.name, t.course_id from faculty f, teaches t where f.id=t.faculty_id) tf,
    -> enrolled e where e.
    -> course_id=c.id and
    -> e.student_id=s.id and
    -> tf.course_id=e.course_id;
+----------------------+--------+-----------------+
| Student              | Course | Faculty         |
+----------------------+--------+-----------------+
| William Hartnell     | CV203  | Carole Ann Ford |
| Paul McGann          | CV203  | Carole Ann Ford |
| Jon Pertwee          | CV385  | Adrienne Hill   |
| Christopher Eccleston | CV385 | Adrienne Hill   |
| Patrick Troughton    | MI101  | Frazer Hines    |
| John Hurt            | MI101  | Frazer Hines    |
| David Tennant        | MI101  | Frazer Hines    |
| Tom Baker            | PH110  | Maureen O'Brien |
| Peter Davison        | ME316  | John Leeson     |
| Sylvestor McCoy      | ME205  | Ian Marter      |
+----------------------+--------+-----------------+
10 rows in set (0.00 sec)
```

7. Display names of courses, faculty teaching them (if any) and course department along with faculty's department

A.
```
select
f.name `Faculty`,
c.name `Course`,
d.name `Faculty Department`,
td.name `Course Department`
from faculty f, courses c, department d,
(
select dd.name, dd.id dept_id, cc.id from department dd, courses cc
where cc.dept_id=dd.id
) td,
teaches tt
```

```
where
tt.course_id=c.id and
tt.faculty_id=f.id and
f.dept_id=d.id and
td.id=c.id and
td.dept_id=c.dept_id;
```

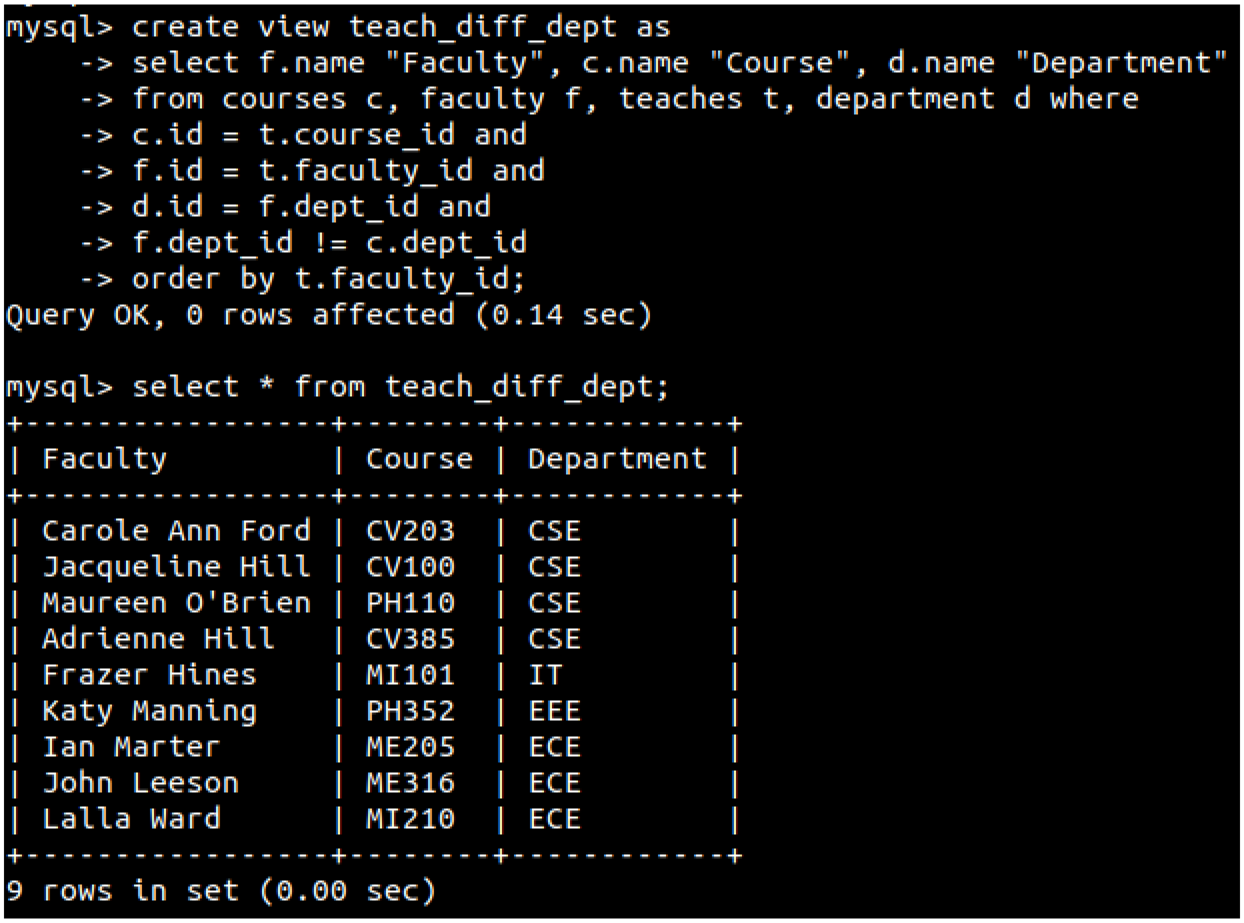


```
mysql> select
    -> f.name `Faculty`,
    -> c.name `Course`,
    -> d.name `Faculty Department`,
    -> td.name `Course Department`
    -> from faculty f, courses c, department d,
    -> (
    -> select dd.name, dd.id dept_id, cc.id from department dd, courses cc
    -> where cc.dept_id=dd.id
    -> ) td,
    -> teaches tt
    -> where
    -> tt.course_id=c.id and
    -> tt.faculty_id=f.id and
    -> f.dept_id=d.id and
    -> td.id=c.id and
    -> td.dept_id=c.dept_id;
+------------------+--------+--------------------+-------------------+
| Faculty          | Course | Faculty Department | Course Department |
+------------------+--------+--------------------+-------------------+
| Jacqueline Hill  | CV100  | CSE                | Civil             |
| Carole Ann Ford  | CV203  | CSE                | Civil             |
| Adrienne Hill    | CV385  | CSE                | Civil             |
| Frazer Hines     | MI101  | IT                 | Mining            |
| Lalla Ward       | MI210  | ECE                | Mining            |
| Katy Manning     | PH352  | EEE                | Phy               |
| Maureen O'Brien  | PH110  | CSE                | Phy               |
| John Leeson      | ME316  | ECE                | Mech              |
| Ian Marter       | ME205  | ECE                | Mech              |
| Wendy Padbury    | IT252  | IT                 | IT                |
+------------------+--------+--------------------+-------------------+
10 rows in set (0.00 sec)
```

Views

1. Display names of all faculty who teach a course that does not belong to their own department

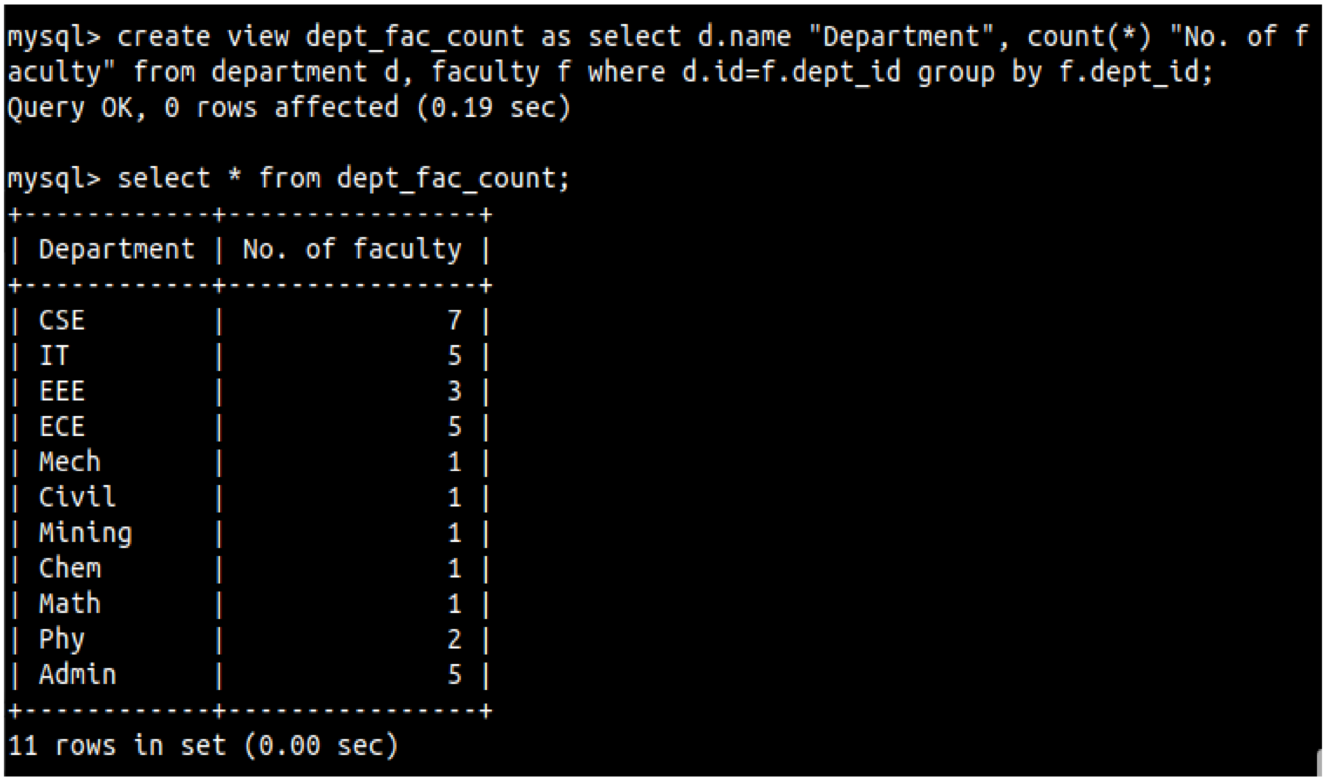
A.
```
create view teach_diff_dept as
select f.name "Faculty", c.name "Course", d.name "Department"
from courses c, faculty f, teaches t, department d where
c.id = t.course_id and
f.id = t.faculty_id and
d.id = f.dept_id and
f.dept_id != c.dept_id
order by t.faculty_id;
```



```
mysql> create view teach_diff_dept as
    -> select f.name "Faculty", c.name "Course", d.name "Department"
    -> from courses c, faculty f, teaches t, department d where
    -> c.id = t.course_id and
    -> f.id = t.faculty_id and
    -> d.id = f.dept_id and
    -> f.dept_id != c.dept_id
    -> order by t.faculty_id;
Query OK, 0 rows affected (0.14 sec)

mysql> select * from teach_diff_dept;
+-----------------+--------+------------+
| Faculty         | Course | Department |
+-----------------+--------+------------+
| Carole Ann Ford | CV203  | CSE        |
| Jacqueline Hill | CV100  | CSE        |
| Maureen O'Brien | PH110  | CSE        |
| Adrienne Hill   | CV385  | CSE        |
| Frazer Hines    | MI101  | IT         |
| Katy Manning    | PH352  | EEE        |
| Ian Marter      | ME205  | ECE        |
| John Leeson     | ME316  | ECE        |
| Lalla Ward      | MI210  | ECE        |
+-----------------+--------+------------+
9 rows in set (0.00 sec)
```

2. Display departments with number of faculty in each department

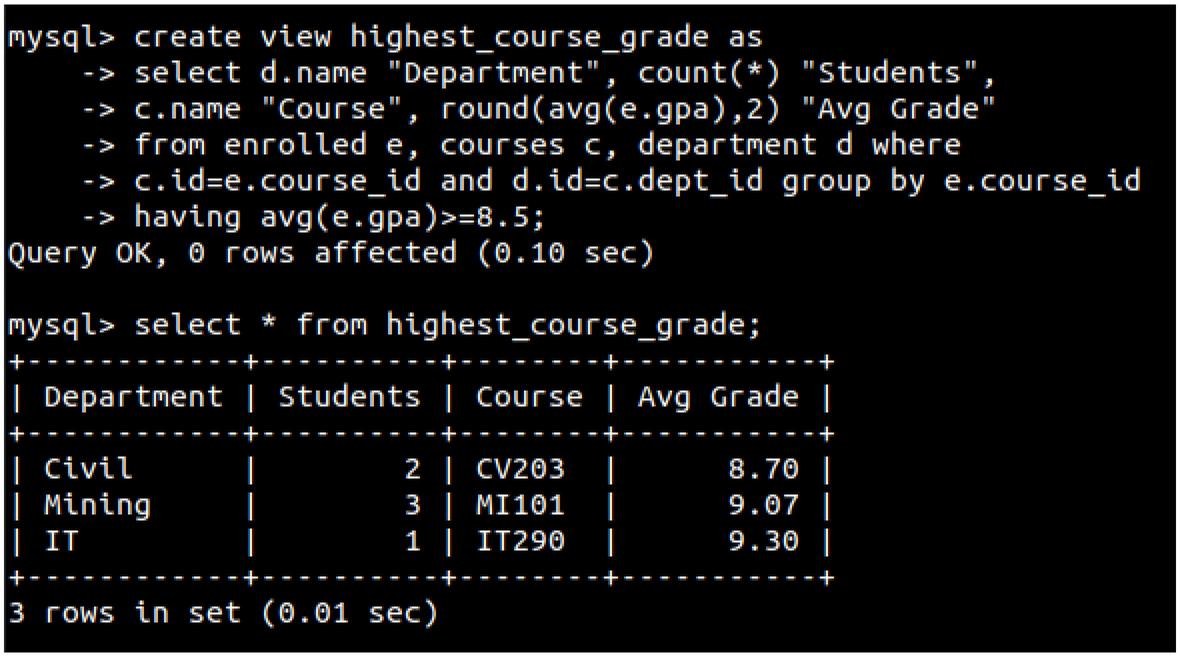
A.
```
create view dept_fac_count as select d.name "Department", count(*)
"No. of faculty" from department d, faculty f where d.id=f.dept_id
group by f.dept_id;
```



```
mysql> create view dept_fac_count as select d.name "Department", count(*) "No. of f
aculty" from department d, faculty f where d.id=f.dept_id group by f.dept_id;
Query OK, 0 rows affected (0.19 sec)

mysql> select * from dept_fac_count;
+------------+----------------+
| Department | No. of faculty |
+------------+----------------+
| CSE        |              7 |
| IT         |              5 |
| EEE        |              3 |
| ECE        |              5 |
| Mech       |              1 |
| Civil      |              1 |
| Mining     |              1 |
| Chem       |              1 |
| Math       |              1 |
| Phy        |              2 |
| Admin      |              5 |
+------------+----------------+
11 rows in set (0.00 sec)
```

3. Display number of students, and department of all courses where the average grade is over 8.5

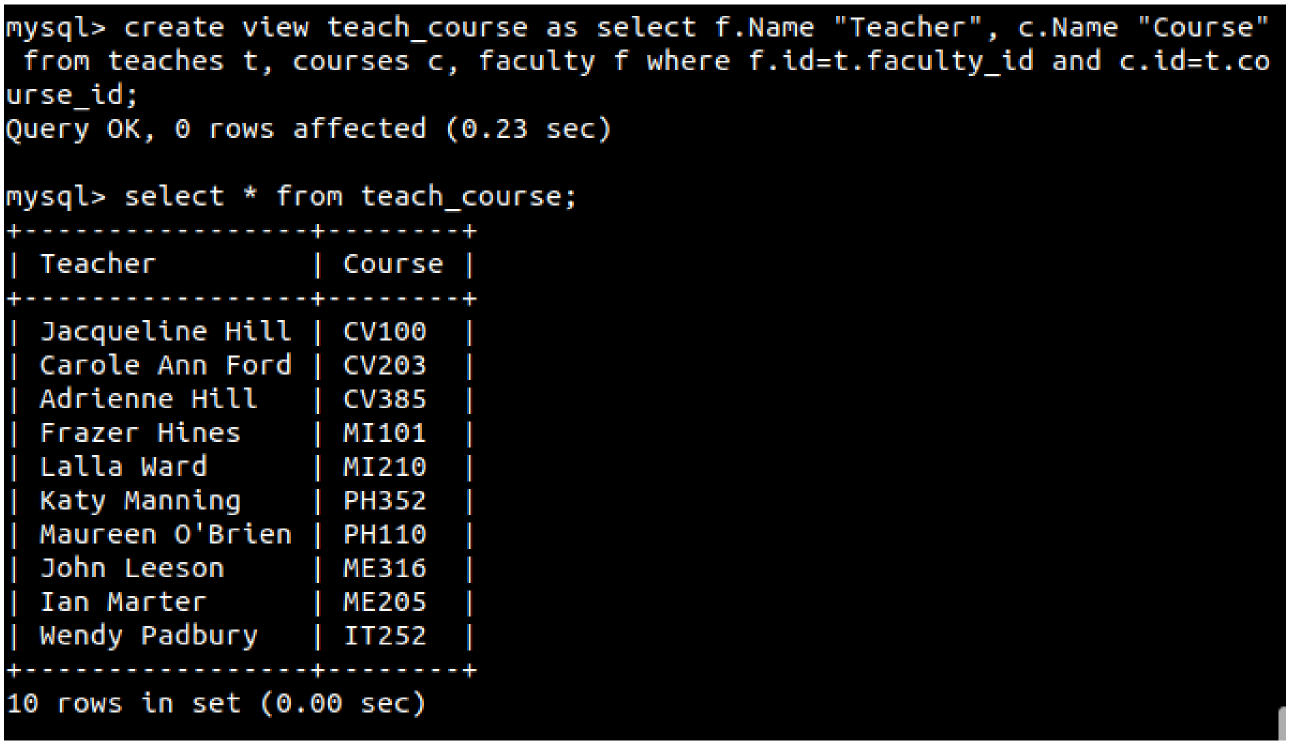
A.
```
create view highest_course_grade as
select d.name "Department", count(*) "Students",
c.name "Course", round(avg(e.gpa),2) "Avg Grade"
from enrolled e, courses c, department d where
c.id=e.course_id and d.id=c.dept_id group by e.course_id
having avg(e.gpa)>=8.5;
```



```
mysql> create view highest_course_grade as
    -> select d.name "Department", count(*) "Students",
    -> c.name "Course", round(avg(e.gpa),2) "Avg Grade"
    -> from enrolled e, courses c, department d where
    -> c.id=e.course_id and d.id=c.dept_id group by e.course_id
    -> having avg(e.gpa)>=8.5;
Query OK, 0 rows affected (0.10 sec)

mysql> select * from highest_course_grade;
+------------+----------+--------+-----------+
| Department | Students | Course | Avg Grade |
+------------+----------+--------+-----------+
| Civil      |        2 | CV203  |      8.70 |
| Mining     |        3 | MI101  |      9.07 |
| IT         |        1 | IT290  |      9.30 |
+------------+----------+--------+-----------+
3 rows in set (0.01 sec)
```

4. Display faculty names along with course they're teaching

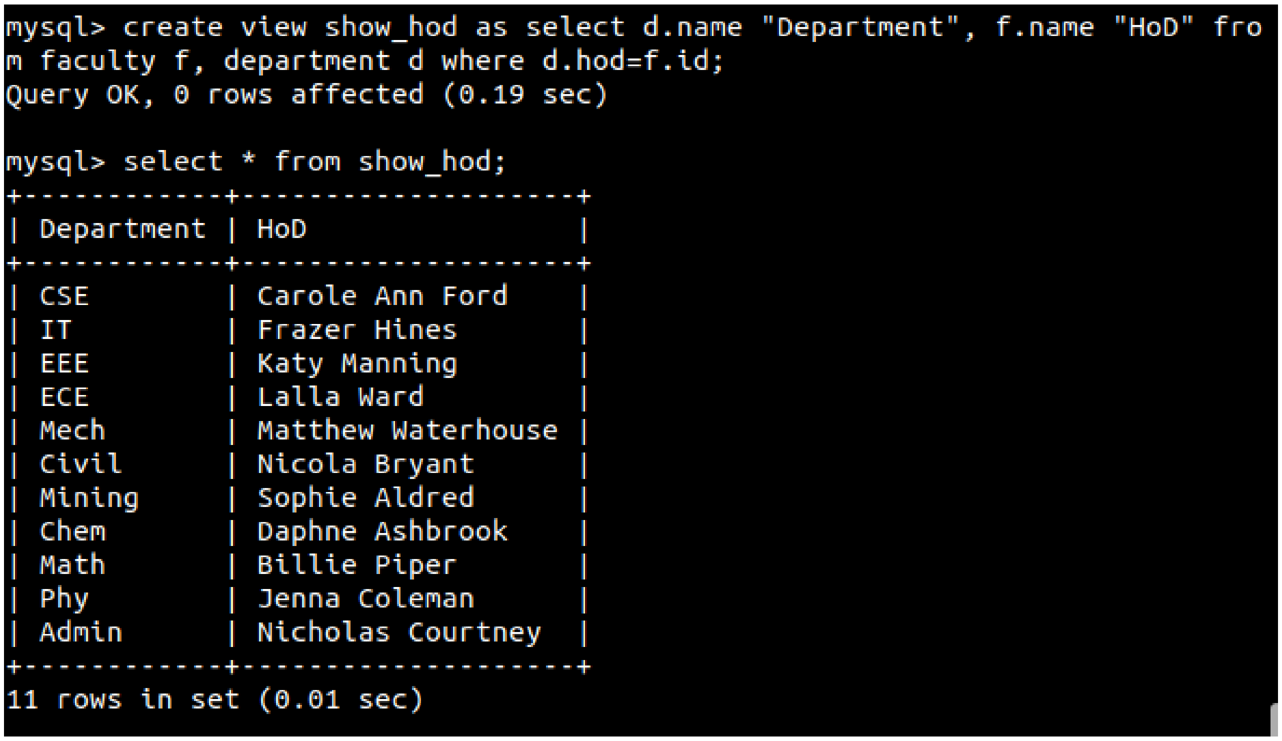
A. `create view teach_course as select f.Name "Teacher", c.Name "Course" from teaches t, courses c, faculty f where f.id=t.faculty_id and c.id=t.course_id;`



```
mysql> create view teach_course as select f.Name "Teacher", c.Name "Course"
 from teaches t, courses c, faculty f where f.id=t.faculty_id and c.id=t.co
urse_id;
Query OK, 0 rows affected (0.23 sec)

mysql> select * from teach_course;
+-----------------+--------+
| Teacher         | Course |
+-----------------+--------+
| Jacqueline Hill | CV100  |
| Carole Ann Ford | CV203  |
| Adrienne Hill   | CV385  |
| Frazer Hines    | MI101  |
| Lalla Ward      | MI210  |
| Katy Manning    | PH352  |
| Maureen O'Brien | PH110  |
| John Leeson     | ME316  |
| Ian Marter      | ME205  |
| Wendy Padbury   | IT252  |
+-----------------+--------+
10 rows in set (0.00 sec)
```

5. Display HoDs of all departments

A. `create view show_hod as select d.name "Department", f.name "HoD" from faculty f, department d where d.hod=f.id;`

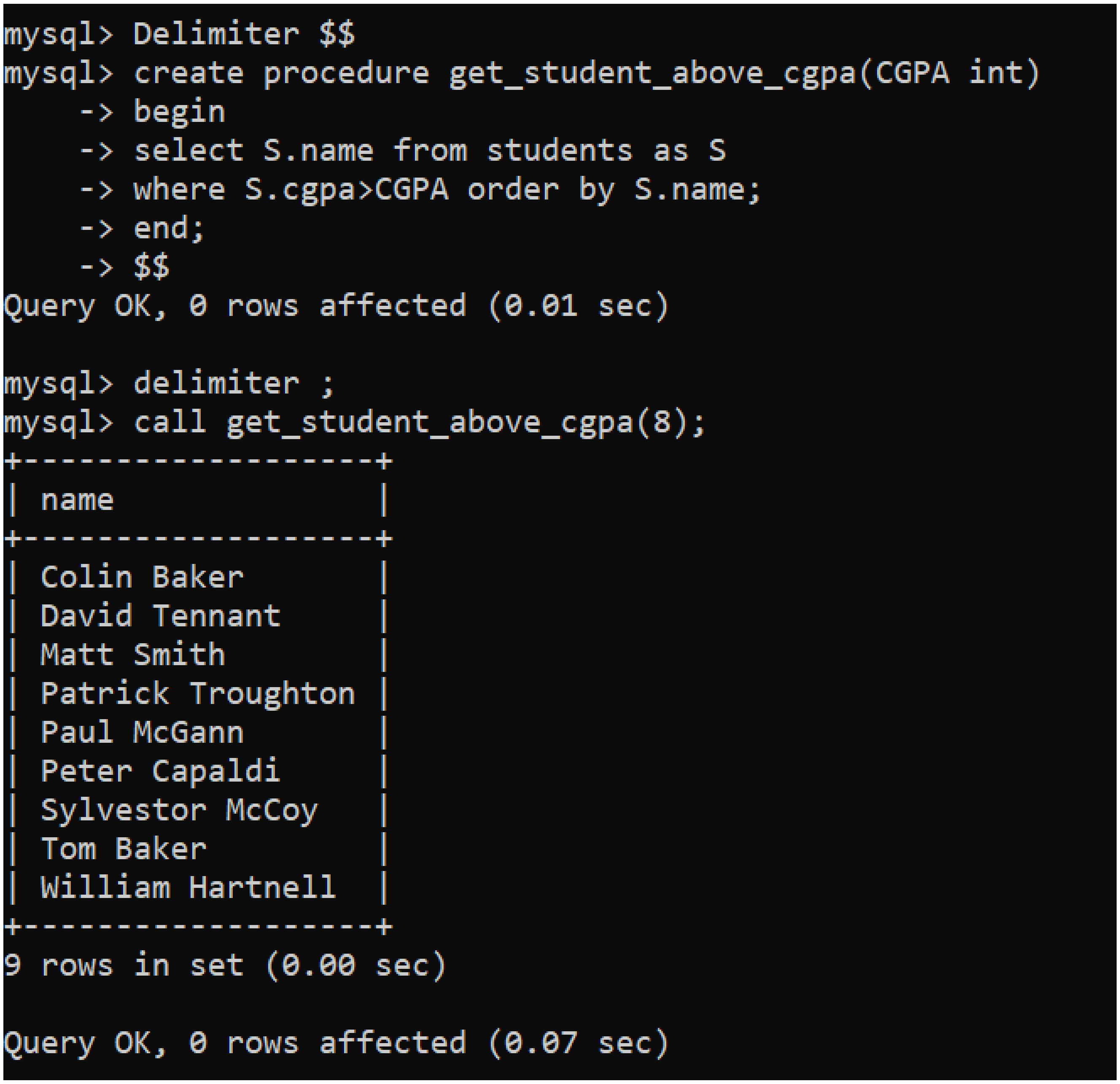


```
mysql> create view show_hod as select d.name "Department", f.name "HoD" fro
m faculty f, department d where d.hod=f.id;
Query OK, 0 rows affected (0.19 sec)

mysql> select * from show_hod;
+------------+--------------------+
| Department | HoD                |
+------------+--------------------+
| CSE        | Carole Ann Ford    |
| IT         | Frazer Hines       |
| EEE        | Katy Manning       |
| ECE        | Lalla Ward         |
| Mech       | Matthew Waterhouse |
| Civil      | Nicola Bryant      |
| Mining     | Sophie Aldred      |
| Chem       | Daphne Ashbrook    |
| Math       | Billie Piper       |
| Phy        | Jenna Coleman      |
| Admin      | Nicholas Courtney  |
+------------+--------------------+
11 rows in set (0.01 sec)
```

Stored Procedures

1. Procedure that accepts CGPA as parameter and displays all student names alphabetically with CGPA above or equal to the given GPA

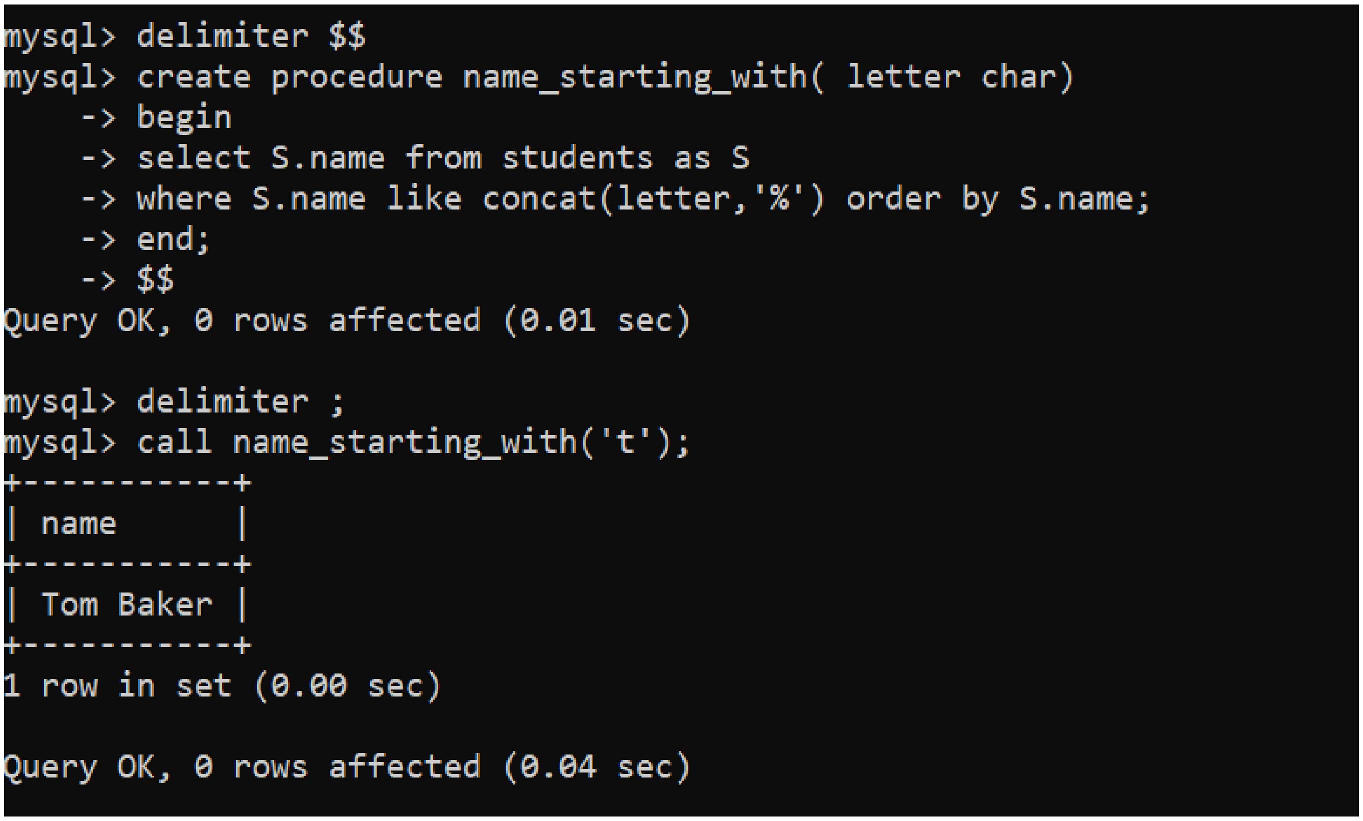
A.
```
delimiter $$
create procedure get_student_above_cgpa(CGPA int)
begin
select S.name from students as S
where S.cgpa>CGPA order by S.name;
end;
$$
delimiter ;
```



```
mysql> Delimiter $$
mysql> create procedure get_student_above_cgpa(CGPA int)
    -> begin
    -> select S.name from students as S
    -> where S.cgpa>CGPA order by S.name;
    -> end;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> call get_student_above_cgpa(8);
+-------------------+
| name              |
+-------------------+
| Colin Baker       |
| David Tennant     |
| Matt Smith        |
| Patrick Troughton |
| Paul McGann       |
| Peter Capaldi     |
| Sylvestor McCoy   |
| Tom Baker         |
| William Hartnell  |
+-------------------+
9 rows in set (0.00 sec)

Query OK, 0 rows affected (0.07 sec)
```

2. Procedure that displays all students names alphabetically starting with a given character

A.
```
delimiter $$
create procedure name_starting_with( letter char)
begin
select S.name from students as S
where S.name like concat(letter,'%') order by S.name;
end;
$$
delimiter ;
```

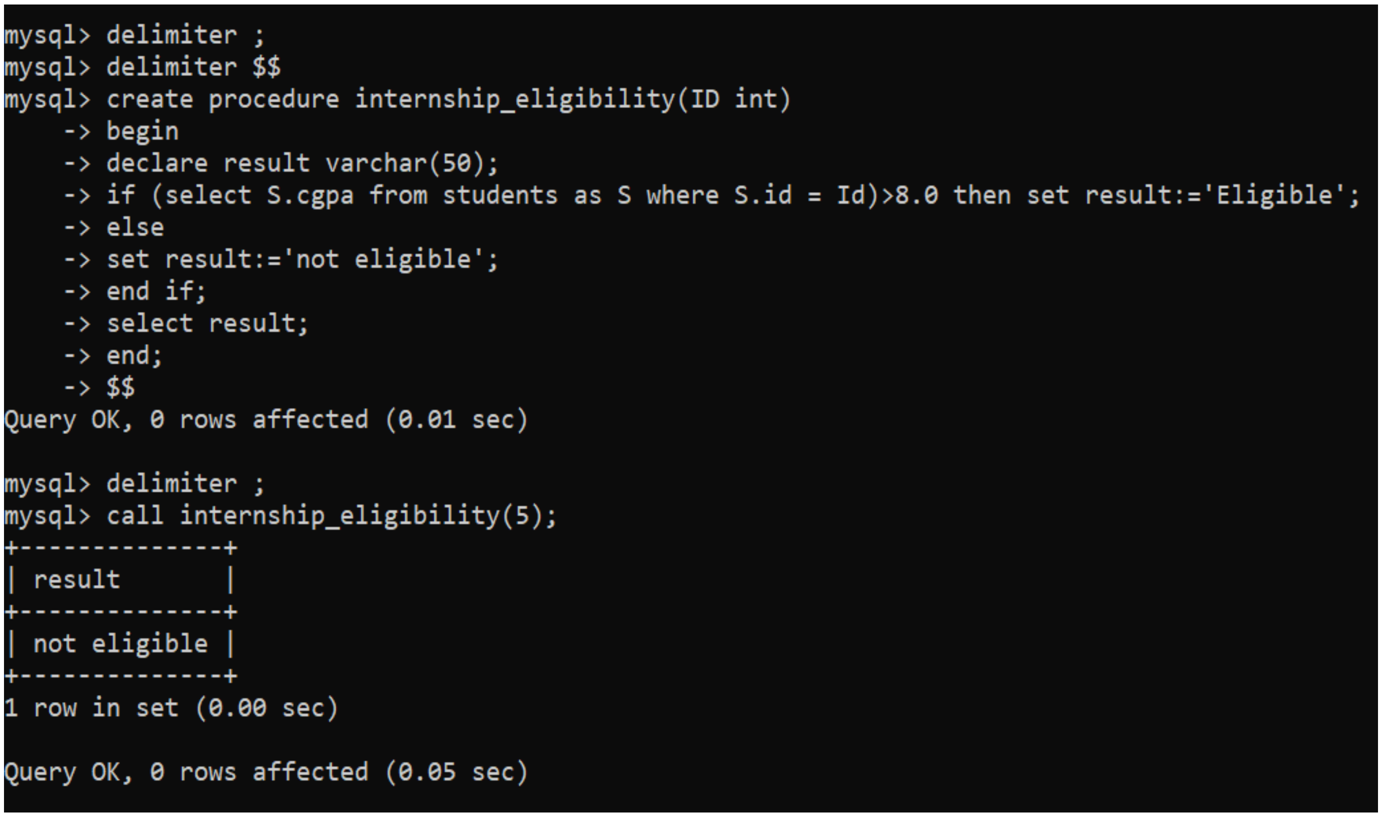


```
mysql> delimiter $$
mysql> create procedure name_starting_with( letter char)
    -> begin
    -> select S.name from students as S
    -> where S.name like concat(letter,'%') order by S.name;
    -> end;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> call name_starting_with('t');
+-----------+
| name      |
+-----------+
| Tom Baker |
+-----------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.04 sec)
```

3. Procedure that takes input as id and tells whether the student is eligible for internship or not (eligibility is 8 cgpa)

A.
```
delimiter $$
create procedure internship_eligibility(ID int)
begin
declare result varchar(50);
if (select S.cgpa from students as S where S.id = Id)>8.0 then set
result:='Eligible';
else
set result:='not eligible';
end if;
select result;
```

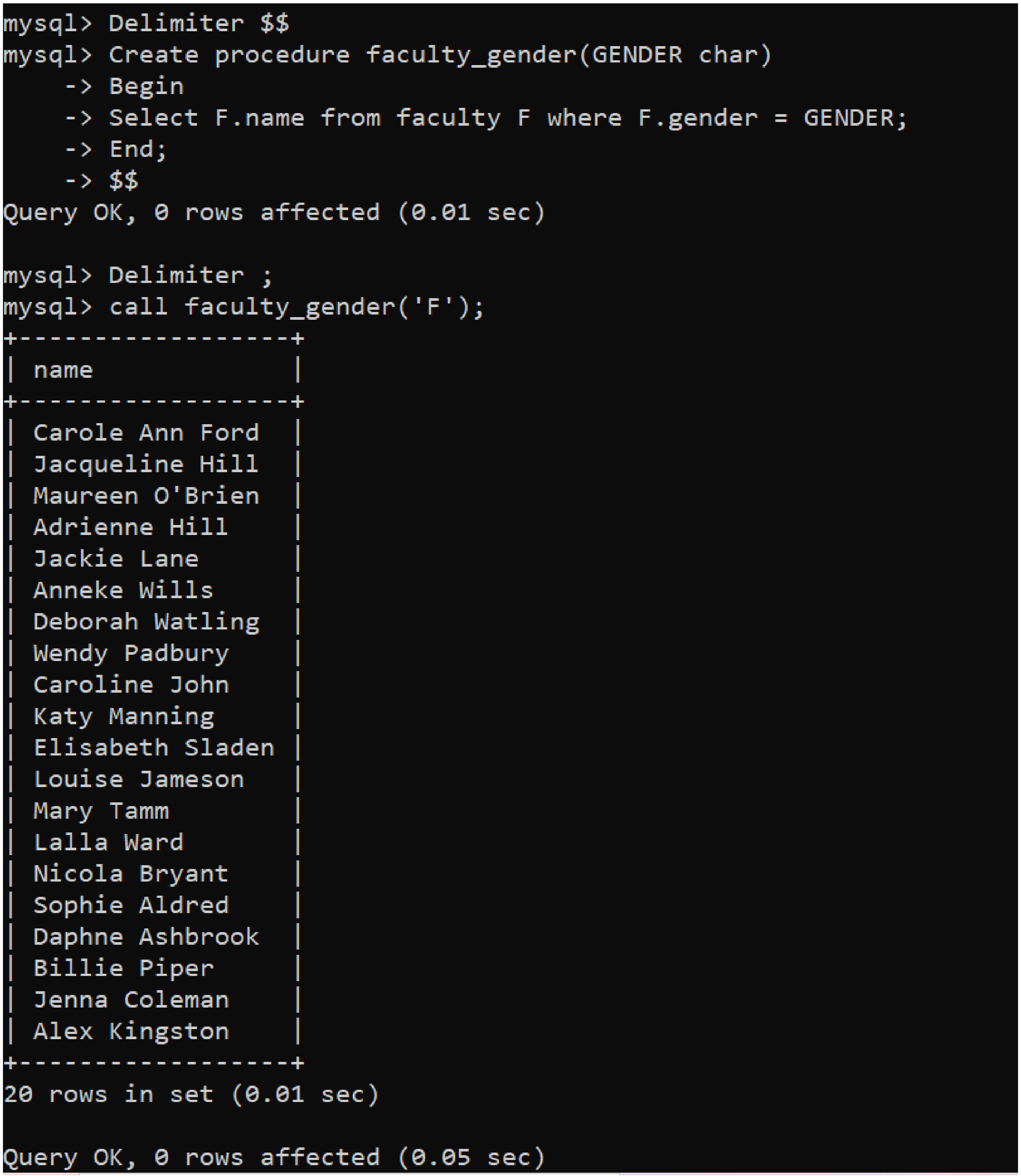
```
end;
$$
delimiter ;
```



```
mysql> delimiter ;
mysql> delimiter $$
mysql> create procedure internship_eligibility(ID int)
    -> begin
    -> declare result varchar(50);
    -> if (select S.cgpa from students as S where S.id = Id)>8.0 then set result:='Eligible';
    -> else
    -> set result:='not eligible';
    -> end if;
    -> select result;
    -> end;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> call internship_eligibility(5);
+--------------+
| result       |
+--------------+
| not eligible |
+--------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.05 sec)
```

4. Procedure that takes gender as input and displays faculty names of that gender.

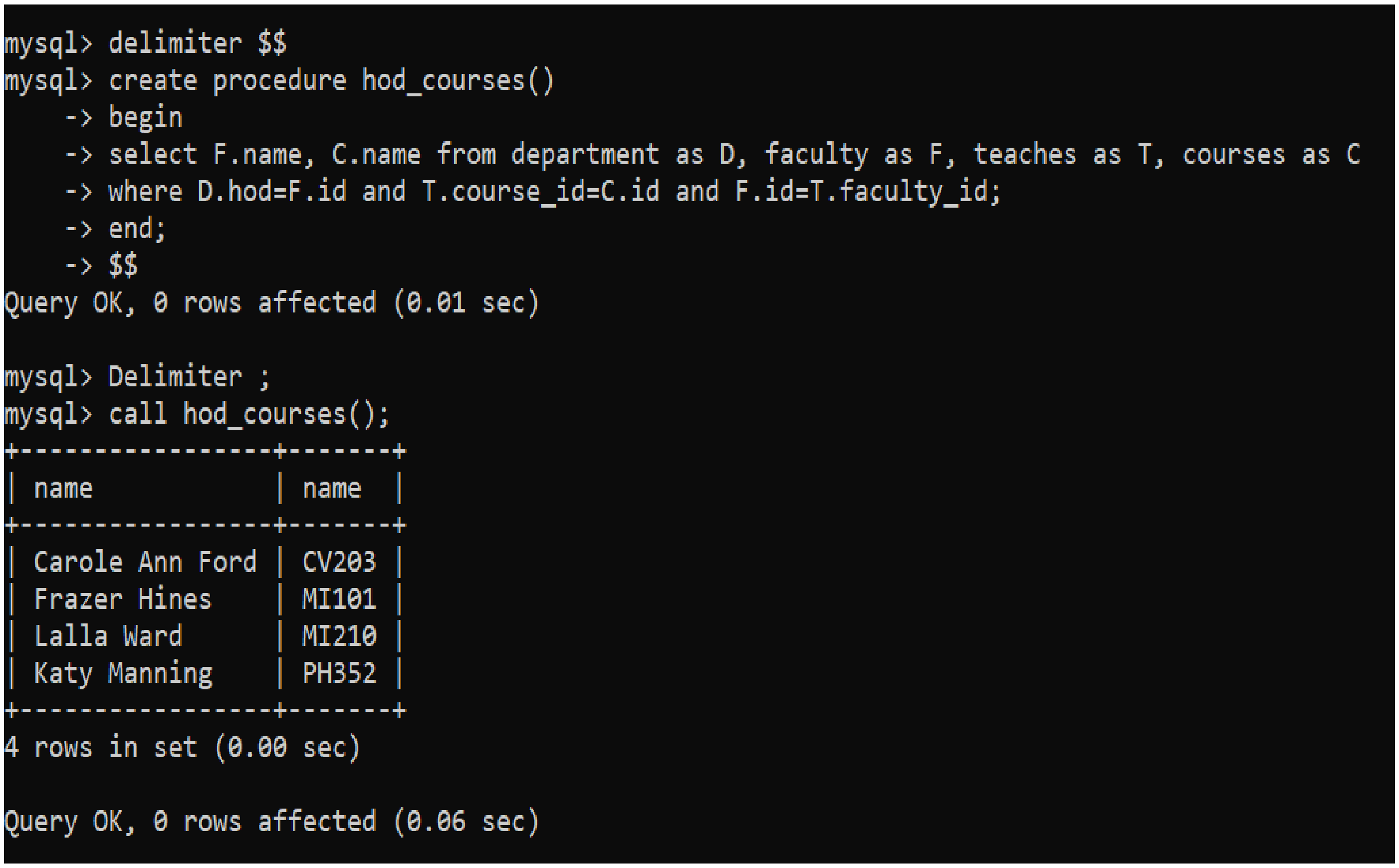
A.
```
delimiter $$
Create procedure faculty_gender(GENDER char)
Begin
Select F.name from faculty F where F.gender = GENDER;
End;
$$
delimiter ;
```



```
mysql> Delimiter $$
mysql> Create procedure faculty_gender(GENDER char)
    -> Begin
    -> Select F.name from faculty F where F.gender = GENDER;
    -> End;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> Delimiter ;
mysql> call faculty_gender('F');
+------------------+
| name             |
+------------------+
| Carole Ann Ford  |
| Jacqueline Hill  |
| Maureen O'Brien  |
| Adrienne Hill    |
| Jackie Lane      |
| Anneke Wills     |
| Deborah Watling  |
| Wendy Padbury    |
| Caroline John    |
| Katy Manning     |
| Elisabeth Sladen |
| Louise Jameson   |
| Mary Tamm        |
| Lalla Ward       |
| Nicola Bryant    |
| Sophie Aldred    |
| Daphne Ashbrook  |
| Billie Piper     |
| Jenna Coleman    |
| Alex Kingston    |
+------------------+
20 rows in set (0.01 sec)

Query OK, 0 rows affected (0.05 sec)
```

5. Procedure that returns the courses taught by the HOD.

A.
```
delimiter $$
create procedure hod_courses()
begin
select F.name, C.name from department as D, faculty as F, teaches as
T, courses as C
where D.hod=F.id and T.course_id=C.id and F.id=T.faculty_id;
end;
$$
delimiter ;
```

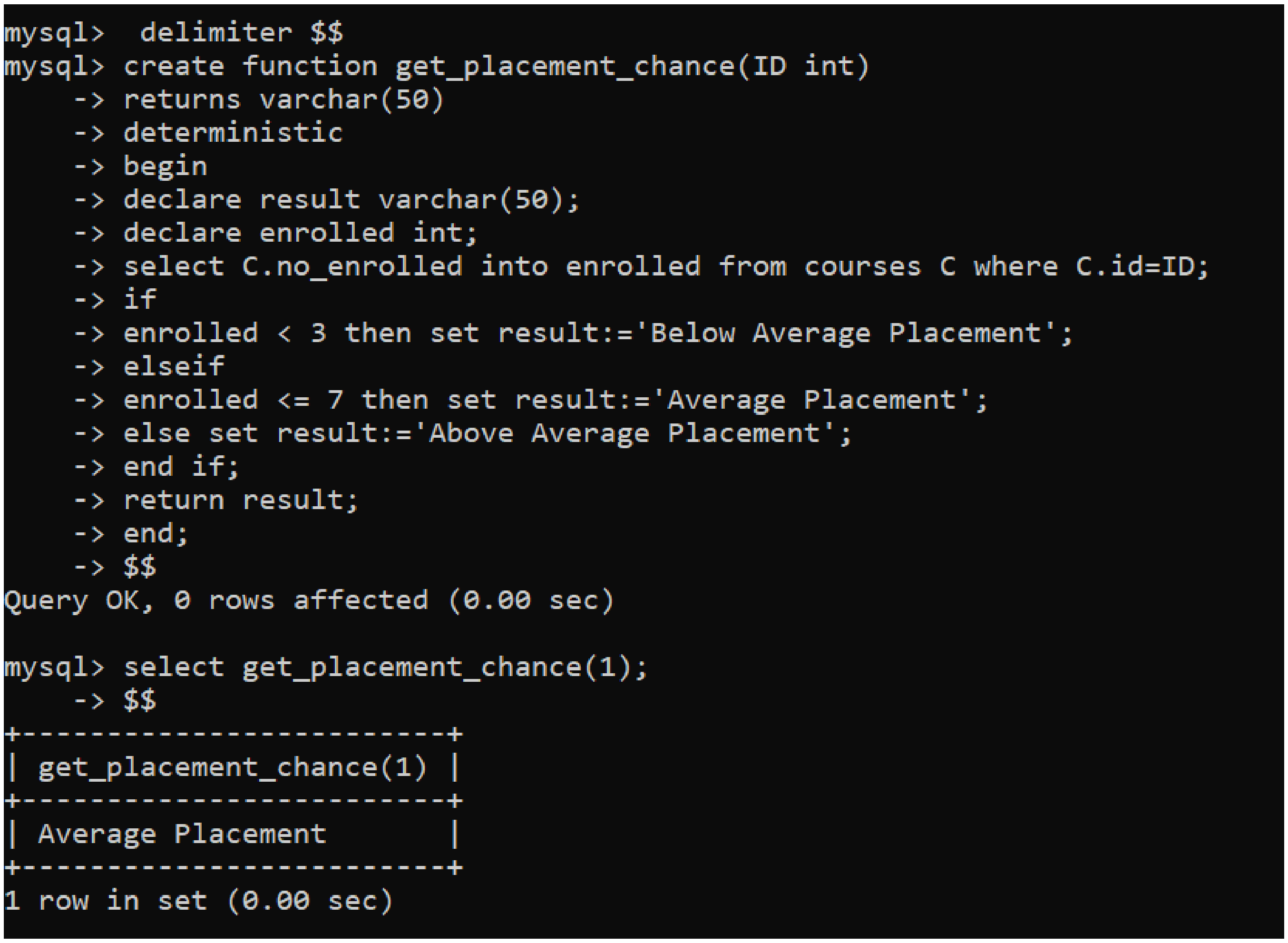


```
mysql> delimiter $$
mysql> create procedure hod_courses()
    -> begin
    -> select F.name, C.name from department as D, faculty as F, teaches as T, courses as C
    -> where D.hod=F.id and T.course_id=C.id and F.id=T.faculty_id;
    -> end;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> Delimiter ;
mysql> call hod_courses();
+-----------------+-------+
| name            | name  |
+-----------------+-------+
| Carole Ann Ford | CV203 |
| Frazer Hines    | MI101 |
| Lalla Ward      | MI210 |
| Katy Manning    | PH352 |
+-----------------+-------+
4 rows in set (0.00 sec)

Query OK, 0 rows affected (0.06 sec)
```

Stored Functions

1. Function that classifies courses based on no_enrolled (highest placement, middle, lowest).

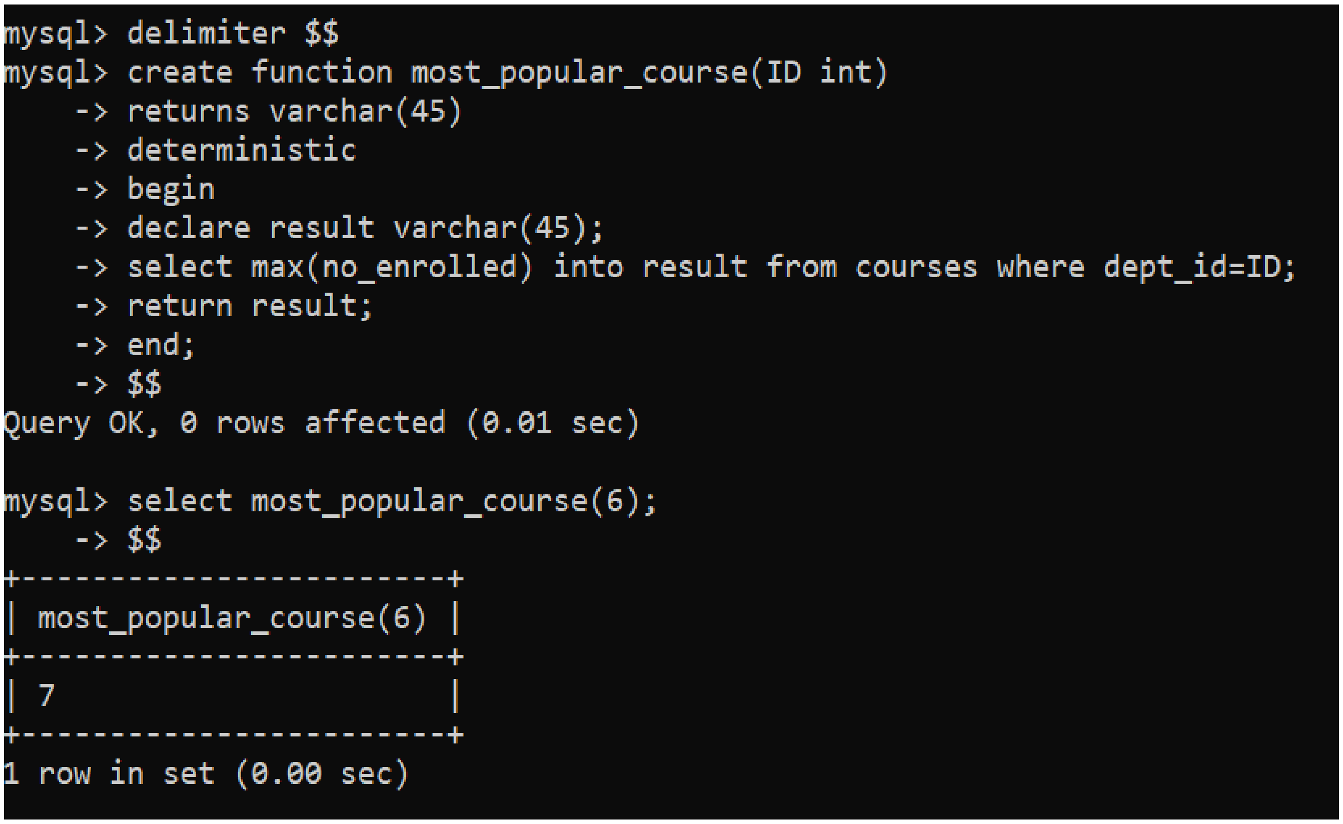
A.
```
delimiter $$
create function get_placement_chance(ID int)
returns varchar(50)
deterministic
begin
declare result varchar(50);
declare enrolled int;
select C.no_enrolled into enrolled from courses C where C.id=ID;
if enrolled < 3 then set result:='Below Average Placement';
elseif enrolled <= 7 then set result:='Average Placement';
else set result:='Above Average Placement';
end if;
return result;
end; $$
Delimiter ;
```



```
mysql>  delimiter $$
mysql> create function get_placement_chance(ID int)
    -> returns varchar(50)
    -> deterministic
    -> begin
    -> declare result varchar(50);
    -> declare enrolled int;
    -> select C.no_enrolled into enrolled from courses C where C.id=ID;
    -> if
    -> enrolled < 3 then set result:='Below Average Placement';
    -> elseif
    -> enrolled <= 7 then set result:='Average Placement';
    -> else set result:='Above Average Placement';
    -> end if;
    -> return result;
    -> end;
    -> $$
Query OK, 0 rows affected (0.00 sec)

mysql> select get_placement_chance(1);
    -> $$
+-------------------------+
| get_placement_chance(1) |
+-------------------------+
| Average Placement       |
+-------------------------+
1 row in set (0.00 sec)
```

2. Function that returns most popular course in each department(input department_id) (based on total enrolled of all courses)

A.
```
delimiter $$
create function most_popular_course(ID int)
returns varchar(45)
deterministic
begin
declare result varchar(45);
select max(no_enrolled) into result from courses where dept_id=ID;
return result;
end;
$$
Delimiter ;
```

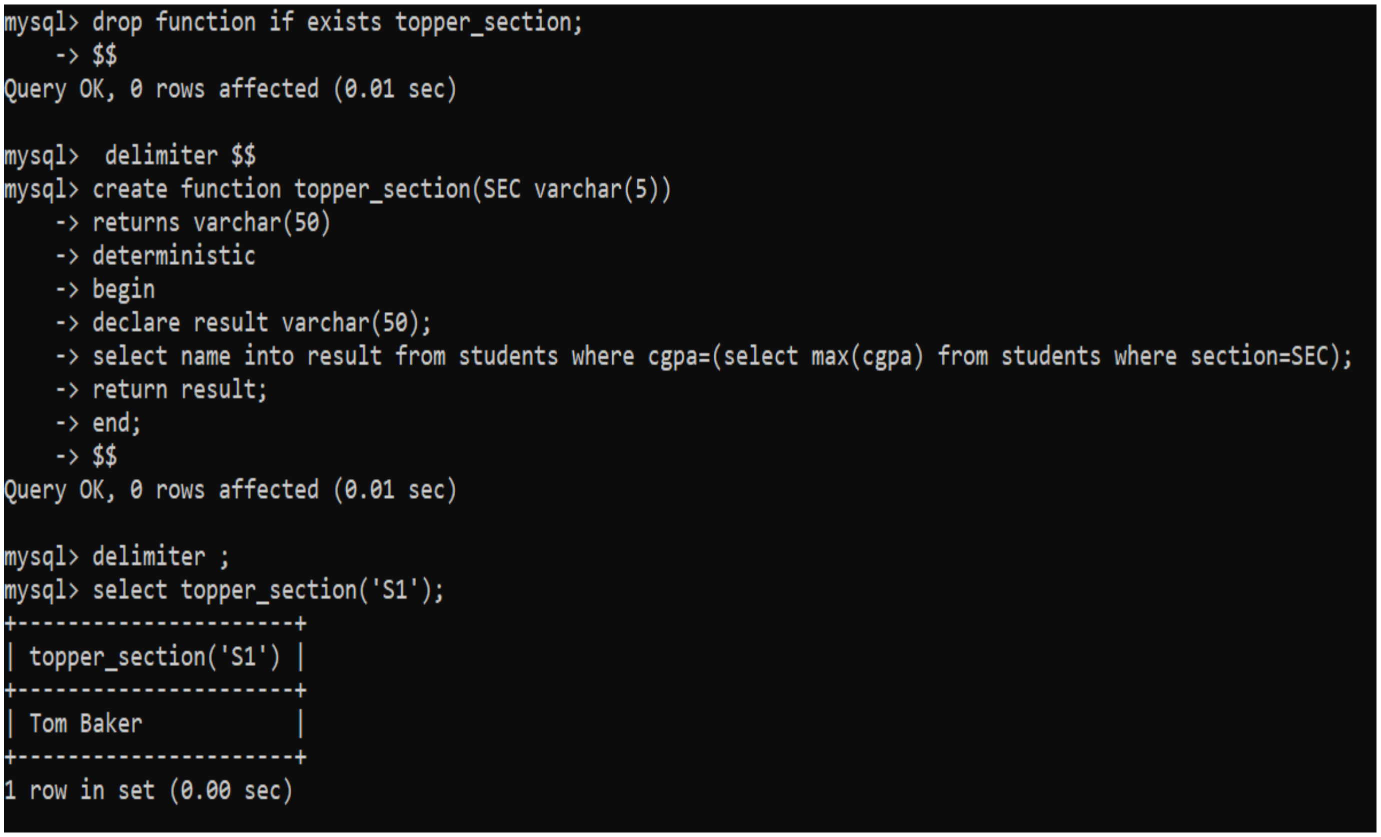


```
mysql> delimiter $$
mysql> create function most_popular_course(ID int)
    -> returns varchar(45)
    -> deterministic
    -> begin
    -> declare result varchar(45);
    -> select max(no_enrolled) into result from courses where dept_id=ID;
    -> return result;
    -> end;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> select most_popular_course(6);
    -> $$
+------------------------+
| most_popular_course(6) |
+------------------------+
| 7                      |
+------------------------+
1 row in set (0.00 sec)
```

3. Function that returns name of the topper from a section given as input

A.
```
delimiter $$
create function topper_section(SEC varchar(5))
returns varchar(50)
deterministic
begin
declare result varchar(50);
select name into result from students where cgpa=(select max(cgpa)
from students where section=SEC);
```

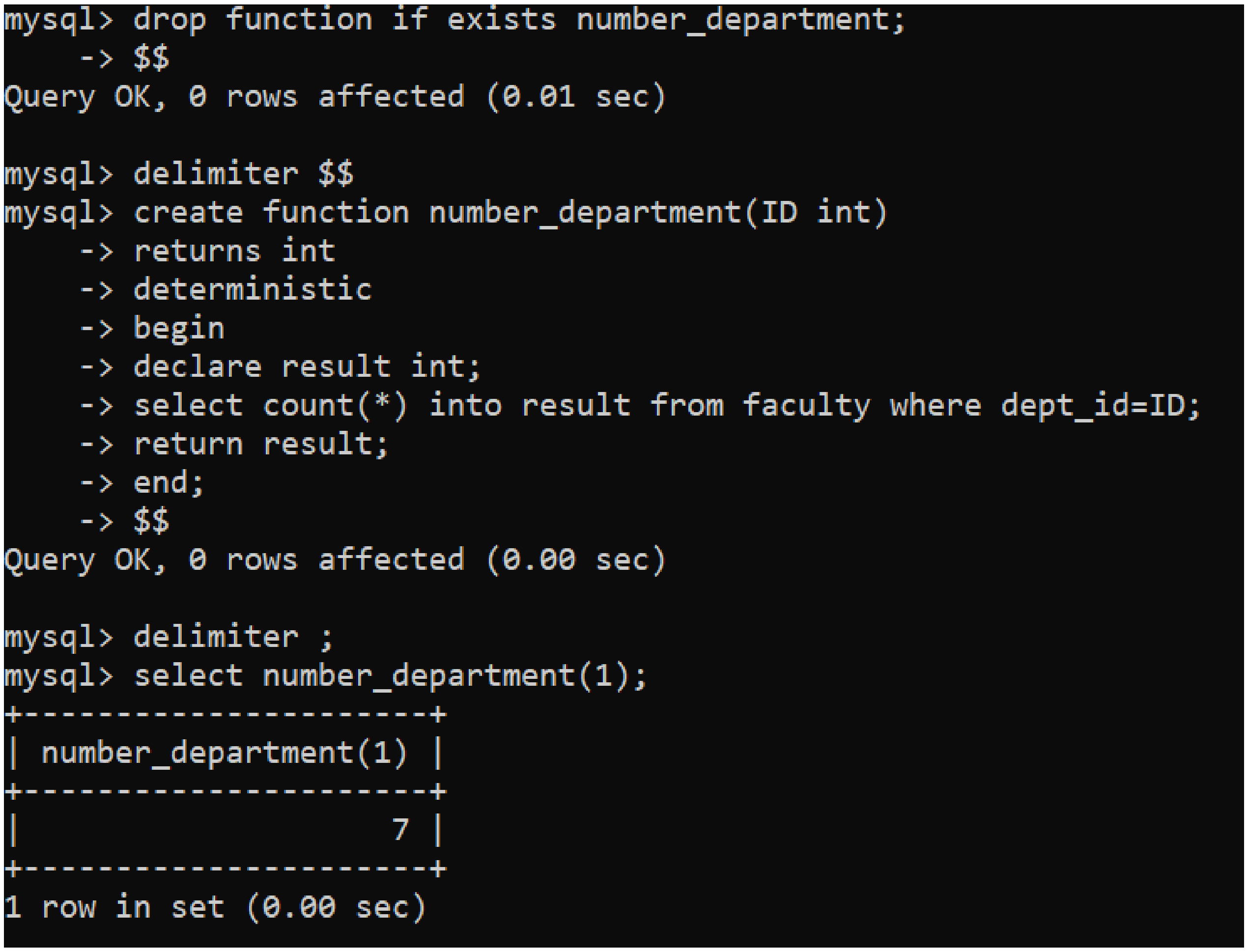
```
return result;
end;
$$
Delimiter ;
```



```
mysql> drop function if exists topper_section;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql>  delimiter $$
mysql> create function topper_section(SEC varchar(5))
    -> returns varchar(50)
    -> deterministic
    -> begin
    -> declare result varchar(50);
    -> select name into result from students where cgpa=(select max(cgpa) from students where section=SEC);
    -> return result;
    -> end;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter ;
mysql> select topper_section('S1');
+----------------------+
| topper_section('S1') |
+----------------------+
| Tom Baker            |
+----------------------+
1 row in set (0.00 sec)
```

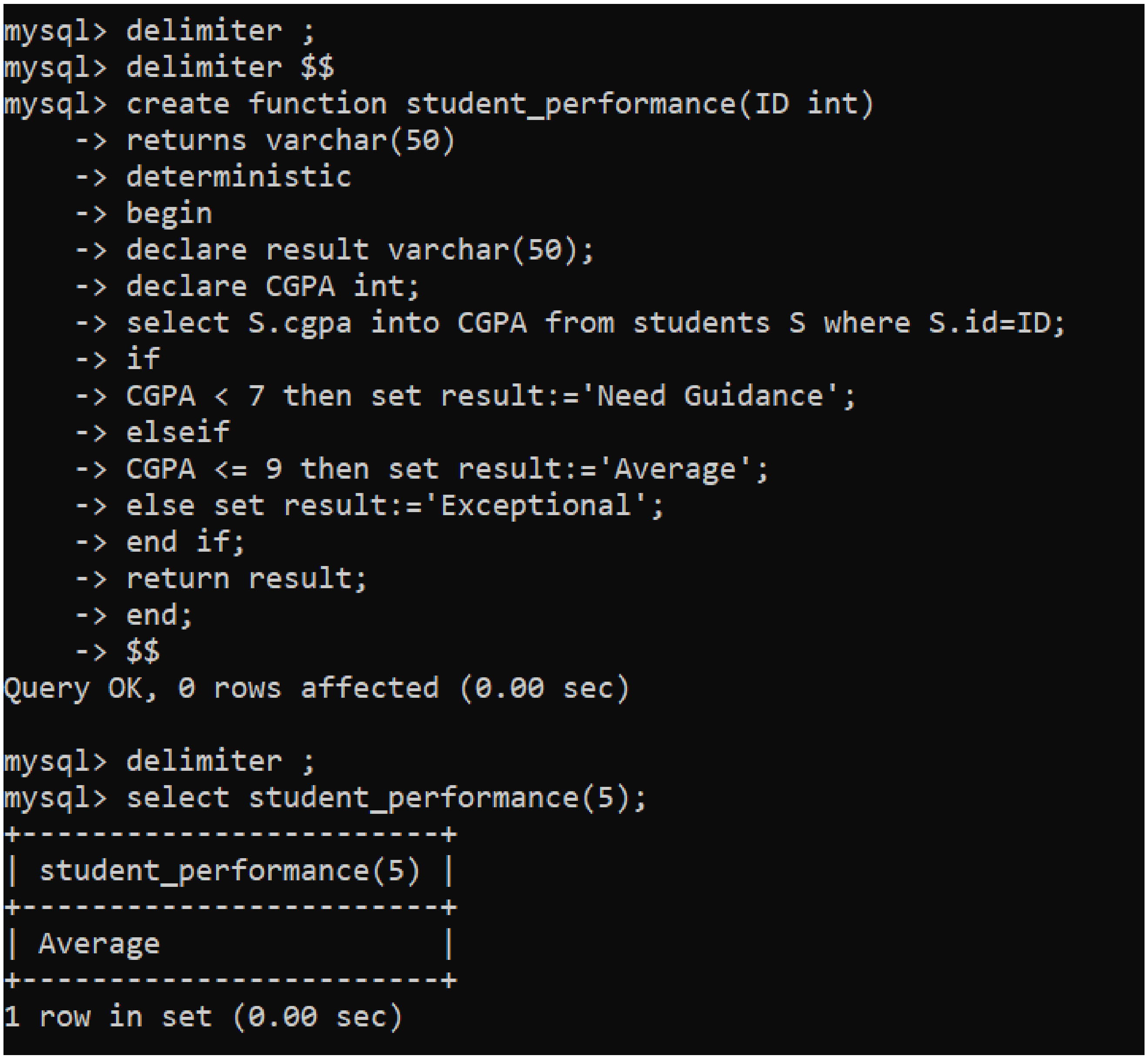
4. **Function that returns number of faculty in the given department (input department_id)**

A.
```
delimiter $$
create function number_department(ID int)
returns int
deterministic
begin
declare result int;
select count(*) into result from faculty where dept_id=ID;
return result;
end;
$$
```


```
mysql> drop function if exists number_department;
    -> $$
Query OK, 0 rows affected (0.01 sec)

mysql> delimiter $$
mysql> create function number_department(ID int)
    -> returns int
    -> deterministic
    -> begin
    -> declare result int;
    -> select count(*) into result from faculty where dept_id=ID;
    -> return result;
    -> end;
    -> $$
Query OK, 0 rows affected (0.00 sec)

mysql> delimiter ;
mysql> select number_department(1);
+----------------------+
| number_department(1) |
+----------------------+
|                    7 |
+----------------------+
1 row in set (0.00 sec)
```

5. Function that classifies the given student as (`Exceptional`, `Average`, `Need Guidance`) based on GPA (input student_id)

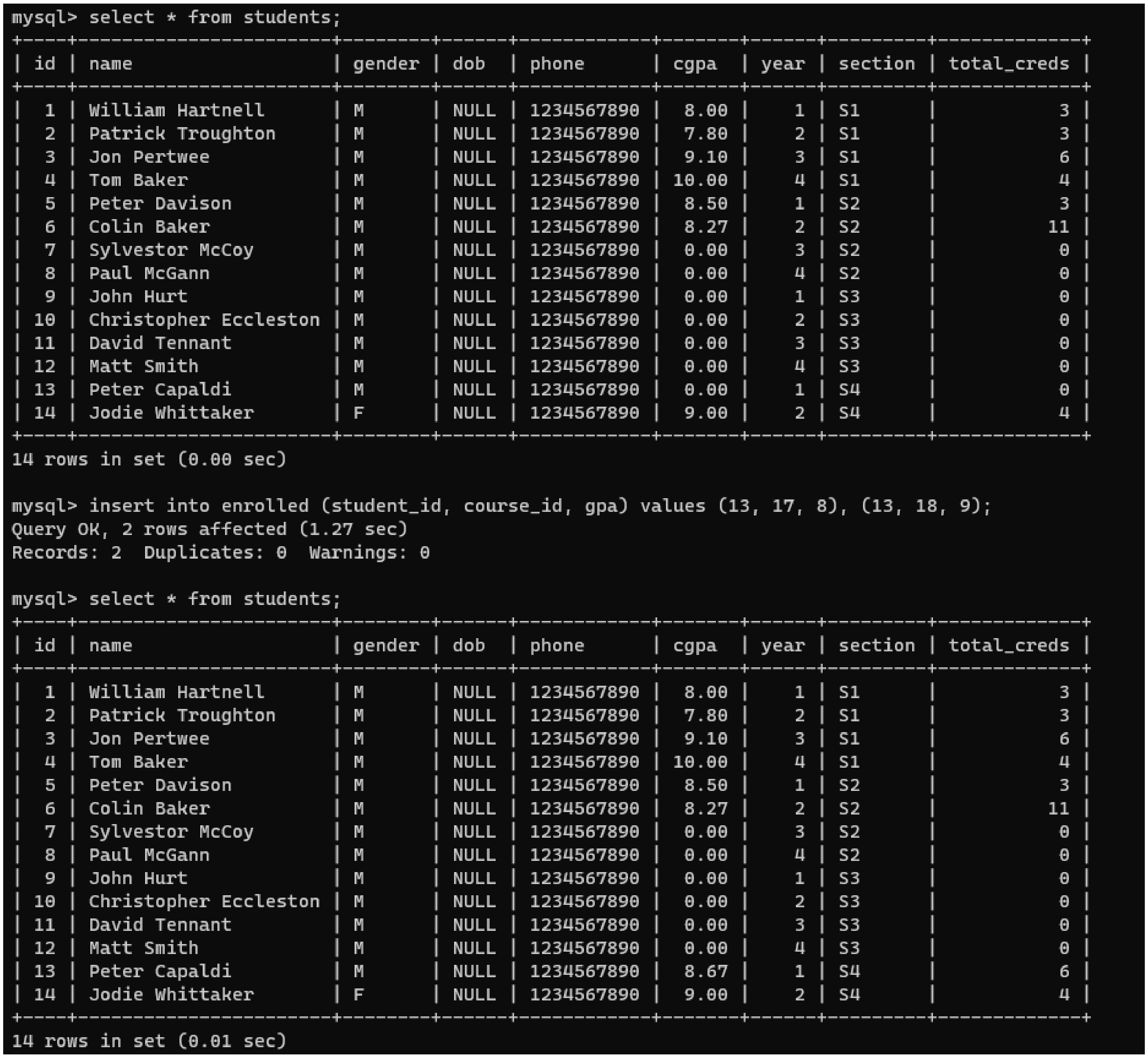
A.
```
delimiter $$
create function student_performance(ID int)
returns varchar(50)
deterministic
begin
declare result varchar(50);
declare CGPA int;
select S.cgpa into CGPA from students S where S.id=ID;
if CGPA < 7 then set result:='Need Guidance';
elseif
CGPA <= 9 then set result:='Average';
else set result:='Exceptional';
end if;
return result;
end;
$$
delimiter ;
```


```
mysql> delimiter ;
mysql> delimiter $$
mysql> create function student_performance(ID int)
    -> returns varchar(50)
    -> deterministic
    -> begin
    -> declare result varchar(50);
    -> declare CGPA int;
    -> select S.cgpa into CGPA from students S where S.id=ID;
    -> if
    -> CGPA < 7 then set result:='Need Guidance';
    -> elseif
    -> CGPA <= 9 then set result:='Average';
    -> else set result:='Exceptional';
    -> end if;
    -> return result;
    -> end;
    -> $$
Query OK, 0 rows affected (0.00 sec)

mysql> delimiter ;
mysql> select student_performance(5);
+------------------------+
| student_performance(5) |
+------------------------+
| Average                |
+------------------------+
1 row in set (0.00 sec)
```

Triggers

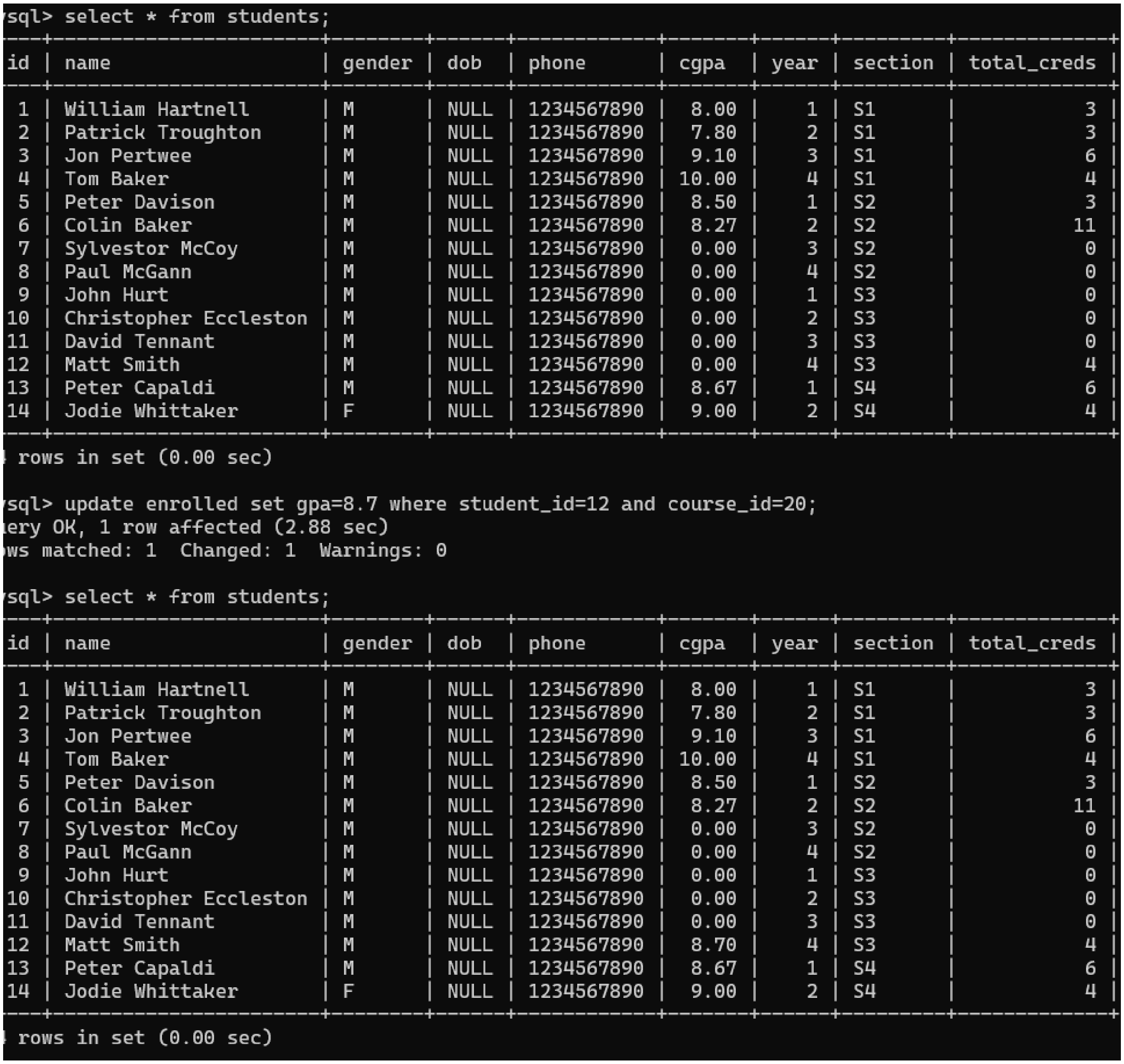
1. Calculating total CGPA - After Insert

A.
```
delimiter $$

create trigger update_gpa
after insert
on enrolled for each row
begin
declare curr_cgpa decimal(5,2) default 0.00;
declare cred int default 0;
declare curr_cred int default 0;
if new.gpa >= 0 then
select cgpa into curr_cgpa from students where id = new.student_id;
select total_creds into curr_cred from students where id =
new.student_id;
select credits into cred from courses where id = new.course_id;
set curr_cgpa = ((curr_cgpa * curr_cred) + (new.gpa *
cred))/(curr_cred + cred);
UPDATE students SET cgpa = curr_cgpa WHERE id = new.student_id;
update students set total_creds = total_creds + cred where id =
new.student_id;
end if;
end $$

delimiter ;
```


```
mysql> select * from students;
+----+-----------------------+--------+------+------------+-------+------+---------+-------------+
| id | name                  | gender | dob  | phone      | cgpa  | year | section | total_creds |
+----+-----------------------+--------+------+------------+-------+------+---------+-------------+
|  1 | William Hartnell      | M      | NULL | 1234567890 |  8.00 |    1 | S1      |           3 |
|  2 | Patrick Troughton     | M      | NULL | 1234567890 |  7.80 |    2 | S1      |           3 |
|  3 | Jon Pertwee           | M      | NULL | 1234567890 |  9.10 |    3 | S1      |           6 |
|  4 | Tom Baker             | M      | NULL | 1234567890 | 10.00 |    4 | S1      |           4 |
|  5 | Peter Davison         | M      | NULL | 1234567890 |  8.50 |    1 | S2      |           3 |
|  6 | Colin Baker           | M      | NULL | 1234567890 |  8.27 |    2 | S2      |          11 |
|  7 | Sylvestor McCoy       | M      | NULL | 1234567890 |  0.00 |    3 | S2      |           0 |
|  8 | Paul McGann           | M      | NULL | 1234567890 |  0.00 |    4 | S2      |           0 |
|  9 | John Hurt             | M      | NULL | 1234567890 |  0.00 |    1 | S3      |           0 |
| 10 | Christopher Eccleston | M      | NULL | 1234567890 |  0.00 |    2 | S3      |           0 |
| 11 | David Tennant         | M      | NULL | 1234567890 |  0.00 |    3 | S3      |           0 |
| 12 | Matt Smith            | M      | NULL | 1234567890 |  0.00 |    4 | S3      |           0 |
| 13 | Peter Capaldi         | M      | NULL | 1234567890 |  0.00 |    1 | S4      |           0 |
| 14 | Jodie Whittaker       | F      | NULL | 1234567890 |  9.00 |    2 | S4      |           4 |
+----+-----------------------+--------+------+------------+-------+------+---------+-------------+
14 rows in set (0.00 sec)

mysql> insert into enrolled (student_id, course_id, gpa) values (13, 17, 8), (13, 18, 9);
Query OK, 2 rows affected (1.27 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from students;
+----+-----------------------+--------+------+------------+-------+------+---------+-------------+
| id | name                  | gender | dob  | phone      | cgpa  | year | section | total_creds |
+----+-----------------------+--------+------+------------+-------+------+---------+-------------+
|  1 | William Hartnell      | M      | NULL | 1234567890 |  8.00 |    1 | S1      |           3 |
|  2 | Patrick Troughton     | M      | NULL | 1234567890 |  7.80 |    2 | S1      |           3 |
|  3 | Jon Pertwee           | M      | NULL | 1234567890 |  9.10 |    3 | S1      |           6 |
|  4 | Tom Baker             | M      | NULL | 1234567890 | 10.00 |    4 | S1      |           4 |
|  5 | Peter Davison         | M      | NULL | 1234567890 |  8.50 |    1 | S2      |           3 |
|  6 | Colin Baker           | M      | NULL | 1234567890 |  8.27 |    2 | S2      |          11 |
|  7 | Sylvestor McCoy       | M      | NULL | 1234567890 |  0.00 |    3 | S2      |           0 |
|  8 | Paul McGann           | M      | NULL | 1234567890 |  0.00 |    4 | S2      |           0 |
|  9 | John Hurt             | M      | NULL | 1234567890 |  0.00 |    1 | S3      |           0 |
| 10 | Christopher Eccleston | M      | NULL | 1234567890 |  0.00 |    2 | S3      |           0 |
| 11 | David Tennant         | M      | NULL | 1234567890 |  0.00 |    3 | S3      |           0 |
| 12 | Matt Smith            | M      | NULL | 1234567890 |  0.00 |    4 | S3      |           0 |
| 13 | Peter Capaldi         | M      | NULL | 1234567890 |  8.67 |    1 | S4      |           6 |
| 14 | Jodie Whittaker       | F      | NULL | 1234567890 |  9.00 |    2 | S4      |           4 |
+----+-----------------------+--------+------+------------+-------+------+---------+-------------+
14 rows in set (0.01 sec)
```

2. Calculating total CGPA - After Update

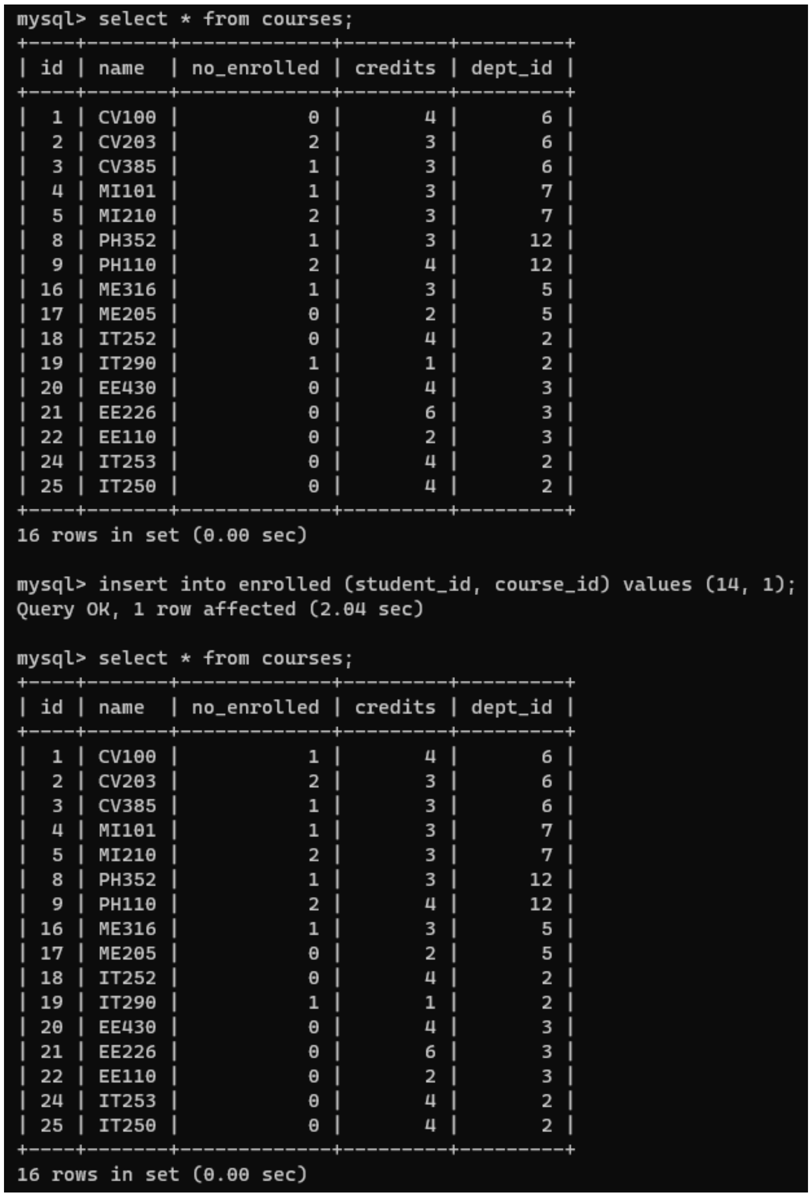
A.
```
delimiter $$

create trigger after_update_gpa
after update
on enrolled for each row
begin
declare curr_cgpa decimal(5,2) default 0.00;
declare cred int default 0;
declare curr_cred int default 0;
if new.gpa >= 0 then
select cgpa into curr_cgpa from students where id = new.student_id;
select total_creds into curr_cred from students where id =
new.student_id;
select credits into cred from courses where id = new.course_id;
set curr_cgpa = ((curr_cgpa * curr_cred) - (old.gpa * cred) +
(new.gpa * cred))/curr_cred;
UPDATE students SET cgpa = curr_cgpa WHERE id = new.student_id;
end if;
end $$

delimiter ;
```



```
ysql> select * from students;
---+----------------------+--------+------+------------+-------+------+---------+-------------+
 id | name                 | gender | dob  | phone      | cgpa  | year | section | total_creds |
---+----------------------+--------+------+------------+-------+------+---------+-------------+
  1 | William Hartnell     | M      | NULL | 1234567890 |  8.00 |    1 | S1      |           3 |
  2 | Patrick Troughton    | M      | NULL | 1234567890 |  7.80 |    2 | S1      |           3 |
  3 | Jon Pertwee          | M      | NULL | 1234567890 |  9.10 |    3 | S1      |           6 |
  4 | Tom Baker            | M      | NULL | 1234567890 | 10.00 |    4 | S1      |           4 |
  5 | Peter Davison        | M      | NULL | 1234567890 |  8.50 |    1 | S2      |           3 |
  6 | Colin Baker          | M      | NULL | 1234567890 |  8.27 |    2 | S2      |          11 |
  7 | Sylvestor McCoy      | M      | NULL | 1234567890 |  0.00 |    3 | S2      |           0 |
  8 | Paul McGann          | M      | NULL | 1234567890 |  0.00 |    4 | S2      |           0 |
  9 | John Hurt            | M      | NULL | 1234567890 |  0.00 |    1 | S3      |           0 |
 10 | Christopher Eccleston | M     | NULL | 1234567890 |  0.00 |    2 | S3      |           0 |
 11 | David Tennant        | M      | NULL | 1234567890 |  0.00 |    3 | S3      |           0 |
 12 | Matt Smith           | M      | NULL | 1234567890 |  0.00 |    4 | S3      |           4 |
 13 | Peter Capaldi        | M      | NULL | 1234567890 |  8.67 |    1 | S4      |           6 |
 14 | Jodie Whittaker      | F      | NULL | 1234567890 |  9.00 |    2 | S4      |           4 |
---+----------------------+--------+------+------------+-------+------+---------+-------------+
4 rows in set (0.00 sec)

ysql> update enrolled set gpa=8.7 where student_id=12 and course_id=20;
ery OK, 1 row affected (2.88 sec)
ows matched: 1  Changed: 1  Warnings: 0

ysql> select * from students;
---+----------------------+--------+------+------------+-------+------+---------+-------------+
 id | name                 | gender | dob  | phone      | cgpa  | year | section | total_creds |
---+----------------------+--------+------+------------+-------+------+---------+-------------+
  1 | William Hartnell     | M      | NULL | 1234567890 |  8.00 |    1 | S1      |           3 |
  2 | Patrick Troughton    | M      | NULL | 1234567890 |  7.80 |    2 | S1      |           3 |
  3 | Jon Pertwee          | M      | NULL | 1234567890 |  9.10 |    3 | S1      |           6 |
  4 | Tom Baker            | M      | NULL | 1234567890 | 10.00 |    4 | S1      |           4 |
  5 | Peter Davison        | M      | NULL | 1234567890 |  8.50 |    1 | S2      |           3 |
  6 | Colin Baker          | M      | NULL | 1234567890 |  8.27 |    2 | S2      |          11 |
  7 | Sylvestor McCoy      | M      | NULL | 1234567890 |  0.00 |    3 | S2      |           0 |
  8 | Paul McGann          | M      | NULL | 1234567890 |  0.00 |    4 | S2      |           0 |
  9 | John Hurt            | M      | NULL | 1234567890 |  0.00 |    1 | S3      |           0 |
 10 | Christopher Eccleston | M     | NULL | 1234567890 |  0.00 |    2 | S3      |           0 |
 11 | David Tennant        | M      | NULL | 1234567890 |  0.00 |    3 | S3      |           0 |
 12 | Matt Smith           | M      | NULL | 1234567890 |  8.70 |    4 | S3      |           4 |
 13 | Peter Capaldi        | M      | NULL | 1234567890 |  8.67 |    1 | S4      |           6 |
 14 | Jodie Whittaker      | F      | NULL | 1234567890 |  9.00 |    2 | S4      |           4 |
---+----------------------+--------+------+------------+-------+------+---------+-------------+
4 rows in set (0.00 sec)
```

3. Increment number of enrolled students after each student enrolls in a course

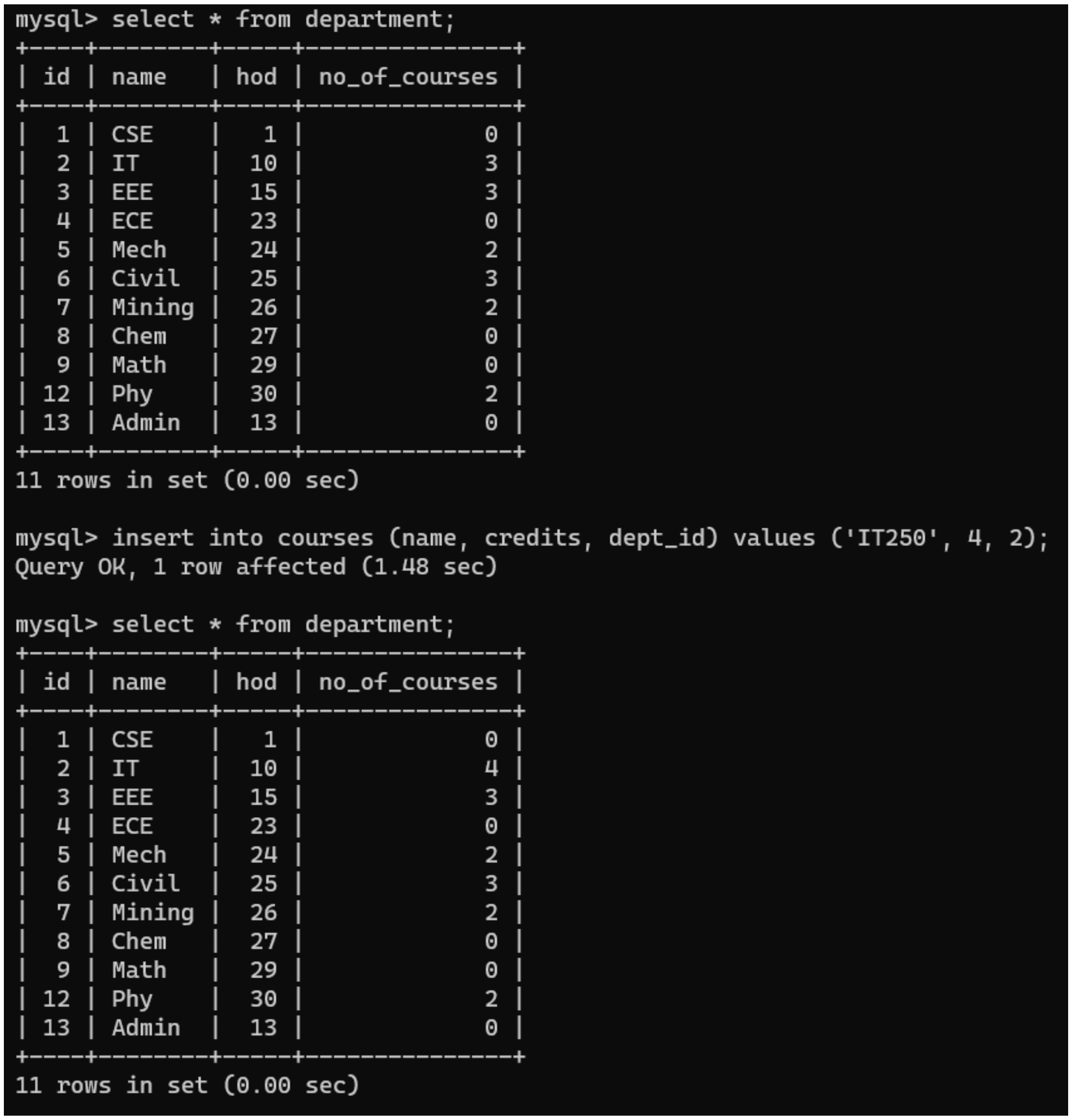
A.
```
delimiter $$

create trigger inc_enrolled_course
after insert
on enrolled for each row
begin
declare curr_enrolled int default 0;
select no_enrolled into curr_enrolled from courses where id =
new.course_id;
UPDATE courses SET no_enrolled = curr_enrolled + 1 WHERE id =
new.course_id;
end $$

delimiter ;
```



```
mysql> select * from courses;
+----+-------+-------------+---------+---------+
| id | name  | no_enrolled | credits | dept_id |
+----+-------+-------------+---------+---------+
|  1 | CV100 |           0 |       4 |       6 |
|  2 | CV203 |           2 |       3 |       6 |
|  3 | CV385 |           1 |       3 |       6 |
|  4 | MI101 |           1 |       3 |       7 |
|  5 | MI210 |           2 |       3 |       7 |
|  8 | PH352 |           1 |       3 |      12 |
|  9 | PH110 |           2 |       4 |      12 |
| 16 | ME316 |           1 |       3 |       5 |
| 17 | ME205 |           0 |       2 |       5 |
| 18 | IT252 |           0 |       4 |       2 |
| 19 | IT290 |           1 |       1 |       2 |
| 20 | EE430 |           0 |       4 |       3 |
| 21 | EE226 |           0 |       6 |       3 |
| 22 | EE110 |           0 |       2 |       3 |
| 24 | IT253 |           0 |       4 |       2 |
| 25 | IT250 |           0 |       4 |       2 |
+----+-------+-------------+---------+---------+
16 rows in set (0.00 sec)

mysql> insert into enrolled (student_id, course_id) values (14, 1);
Query OK, 1 row affected (2.04 sec)

mysql> select * from courses;
+----+-------+-------------+---------+---------+
| id | name  | no_enrolled | credits | dept_id |
+----+-------+-------------+---------+---------+
|  1 | CV100 |           1 |       4 |       6 |
|  2 | CV203 |           2 |       3 |       6 |
|  3 | CV385 |           1 |       3 |       6 |
|  4 | MI101 |           1 |       3 |       7 |
|  5 | MI210 |           2 |       3 |       7 |
|  8 | PH352 |           1 |       3 |      12 |
|  9 | PH110 |           2 |       4 |      12 |
| 16 | ME316 |           1 |       3 |       5 |
| 17 | ME205 |           0 |       2 |       5 |
| 18 | IT252 |           0 |       4 |       2 |
| 19 | IT290 |           1 |       1 |       2 |
| 20 | EE430 |           0 |       4 |       3 |
| 21 | EE226 |           0 |       6 |       3 |
| 22 | EE110 |           0 |       2 |       3 |
| 24 | IT253 |           0 |       4 |       2 |
| 25 | IT250 |           0 |       4 |       2 |
+----+-------+-------------+---------+---------+
16 rows in set (0.00 sec)
```

4. Increment number of courses in department after insert in courses

A.
```
delimiter $$

create trigger courses_after_insert
after insert
on courses for each row
begin
update department set no_of_courses = no_of_courses + 1 where id =
new.dept_id;
end $$

delimiter ;
```



```
mysql> select * from department;
+----+--------+-----+---------------+
| id | name   | hod | no_of_courses |
+----+--------+-----+---------------+
|  1 | CSE    |   1 |             0 |
|  2 | IT     |  10 |             3 |
|  3 | EEE    |  15 |             3 |
|  4 | ECE    |  23 |             0 |
|  5 | Mech   |  24 |             2 |
|  6 | Civil  |  25 |             3 |
|  7 | Mining |  26 |             2 |
|  8 | Chem   |  27 |             0 |
|  9 | Math   |  29 |             0 |
| 12 | Phy    |  30 |             2 |
| 13 | Admin  |  13 |             0 |
+----+--------+-----+---------------+
11 rows in set (0.00 sec)

mysql> insert into courses (name, credits, dept_id) values ('IT250', 4, 2);
Query OK, 1 row affected (1.48 sec)

mysql> select * from department;
+----+--------+-----+---------------+
| id | name   | hod | no_of_courses |
+----+--------+-----+---------------+
|  1 | CSE    |   1 |             0 |
|  2 | IT     |  10 |             4 |
|  3 | EEE    |  15 |             3 |
|  4 | ECE    |  23 |             0 |
|  5 | Mech   |  24 |             2 |
|  6 | Civil  |  25 |             3 |
|  7 | Mining |  26 |             2 |
|  8 | Chem   |  27 |             0 |
|  9 | Math   |  29 |             0 |
| 12 | Phy    |  30 |             2 |
| 13 | Admin  |  13 |             0 |
+----+--------+-----+---------------+
11 rows in set (0.00 sec)
```