

CS313 - Lab 5

Q.1

Q2. Find top 5 students (get their names and department) by tot_cred (i.e, those top 5 who have completed highest total credits).

(This question needs to be submitted with query and output).

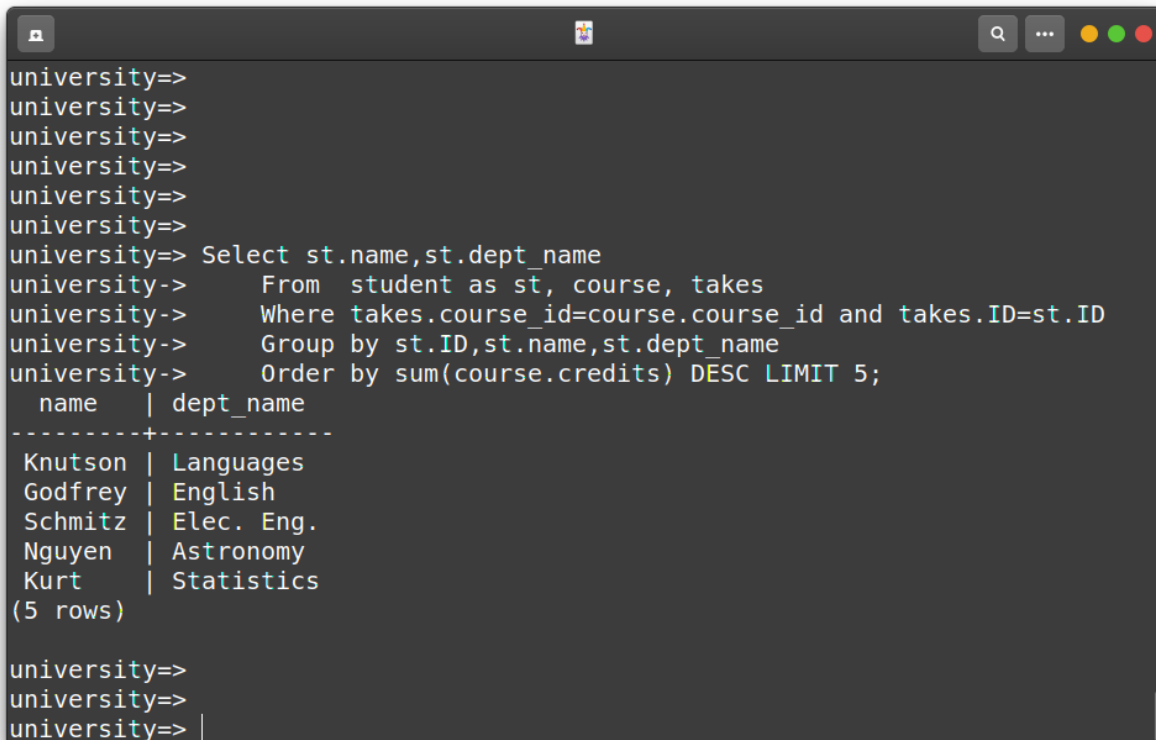
Select st.name,st.dept_name

From student as st, course, takes

Where takes.course_id=course.course_id and takes.ID=st.ID

Group by st.ID,st.name,st.dept_name

Order by sum(course.credits) DESC LIMIT 5;



```
university=>
university=>
university=>
university=>
university=>
university=>
university=> Select st.name,st.dept_name
university->      From student as st, course, takes
university->      Where takes.course_id=course.course_id and takes.ID=st.ID
university->      Group by st.ID,st.name,st.dept_name
university->      Order by sum(course.credits) DESC LIMIT 5;
  name  | dept_name
-----+-----
Knutson | Languages
Godfrey | English
Schmitz | Elec. Eng.
Nguyen  | Astronomy
Kurt    | Statistics
(5 rows)

university=>
university=>
university=> |
```

Q.2

Q5. Use begin transaction SQL statement along with commit/rollback.

(This question needs to be submitted with a query or statements of sql file and output)

You need to write insert statements for the following:

- i) add a course
- ii) create a section for it within some year/semester etc.
- iii) assign a teacher to it

Create a file (.sql type) containing the above statements with begin,commit or rollback transactions as per the following cases.Create separate .sql file for each of the following cases and execute :

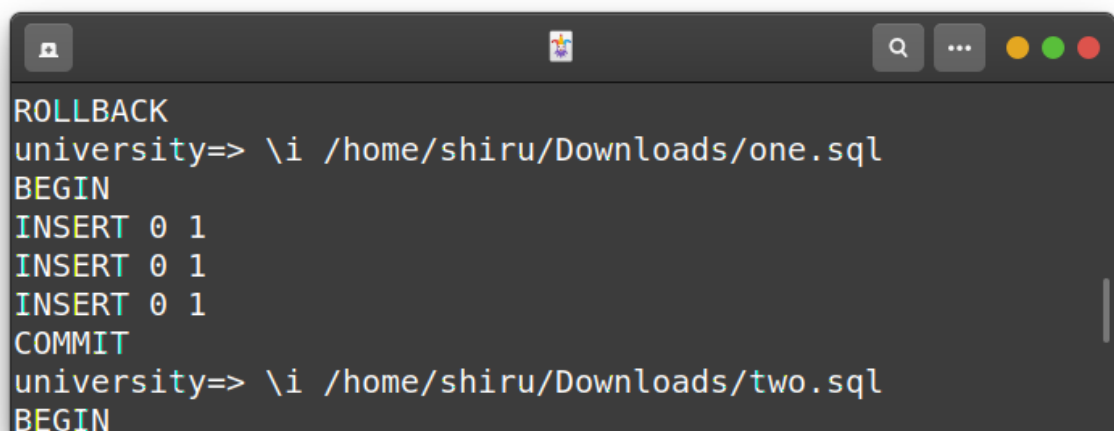
Begin TRANSACTION;

Insert into course VALUES ('433', 'Potterology', 'Comp. Sci.', 4);

Insert into section VALUES (433, '1', 'Spring', '2010', 'Whitman', '434', 'O');

Insert into teaches VALUES ('3335', 433, '1', 'Spring', '2010');

Commit TRANSACTION;



```
ROLLBACK
university=> \i /home/shiru/Downloads/one.sql
BEGIN
INSERT 0 1
INSERT 0 1
INSERT 0 1
COMMIT
university=> \i /home/shiru/Downloads/two.sql
BEGIN
```

Begin TRANSACTION;
Insert into course VALUES (433, 'Potterology', 'Comp. Sci.', 4);
Insert into section VALUES (433, '1', 'Spring', '2010', 'Whitman', '434', 'O');
Rollback TRANSACTION;
Insert into teaches VALUES ('3335', 433, '1', 'Spring', '2010');
Commit TRANSACTION;

```
COMMIT
university=> \i /home/shiru/Downloads/two.sql
BEGIN
INSERT 0 1
INSERT 0 1
ROLLBACK
psql:/home/shiru/Downloads/two.sql:8: ERROR:  insert or update on table "teaches" violates
foreign key constraint "teaches_course_id_sec_id_semester_year_fkey"
DETAIL:  Key (course_id, sec_id, semester, year)=(433, 1, Spring, 2010) is not present in t
able "section".
psql:/home/shiru/Downloads/two.sql:9: WARNING:  there is no transaction in progress
COMMIT
```

Begin TRANSACTION;
Insert into course VALUES ('433', 'Weasleyology', 'Comp. Sci.', 4);
Insert into section VALUES ('433', '1', 'Spring', '2010', 'Whitman', '134', 'O');
Insert into teaches VALUES ('3335', '433', '1', 'Spring', '2010');
Rollback TRANSACTION;

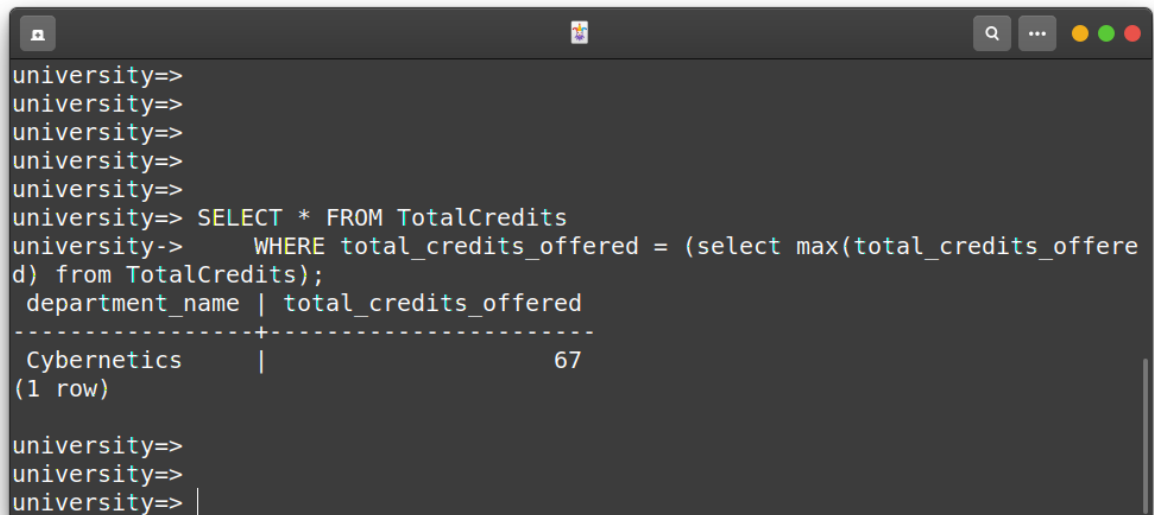
```
university=>
university=>
university=>
university=>
university=>
university=>
university=> \i /home/shiru/Downloads/three.sql
BEGIN
INSERT 0 1
INSERT 0 1
INSERT 0 1
ROLLBACK
university=>
university=>
university=> |
```

Q.3

```
CREATE VIEW TotalCredits AS
  SELECT department.dept_name AS department_name, sum(course.credits)
AS total_credits_offered
  FROM department,course WHERE  department.dept_name =
course.dept_name
  group by department.dept_name;
```

Output : CREATE VIEW

```
SELECT * FROM TotalCredits
  WHERE total_credits_offered = (select max(total_credits_offered) from
TotalCredits);
```



The screenshot shows a terminal window with a dark background. The prompt is 'university=>'. The user enters the SQL command: 'SELECT * FROM TotalCredits WHERE total_credits_offered = (select max(total_credits_offered) from TotalCredits);'. The output shows a single row with 'Cybernetics' as the department name and '67' as the total credits offered. The prompt then returns to 'university=>'.

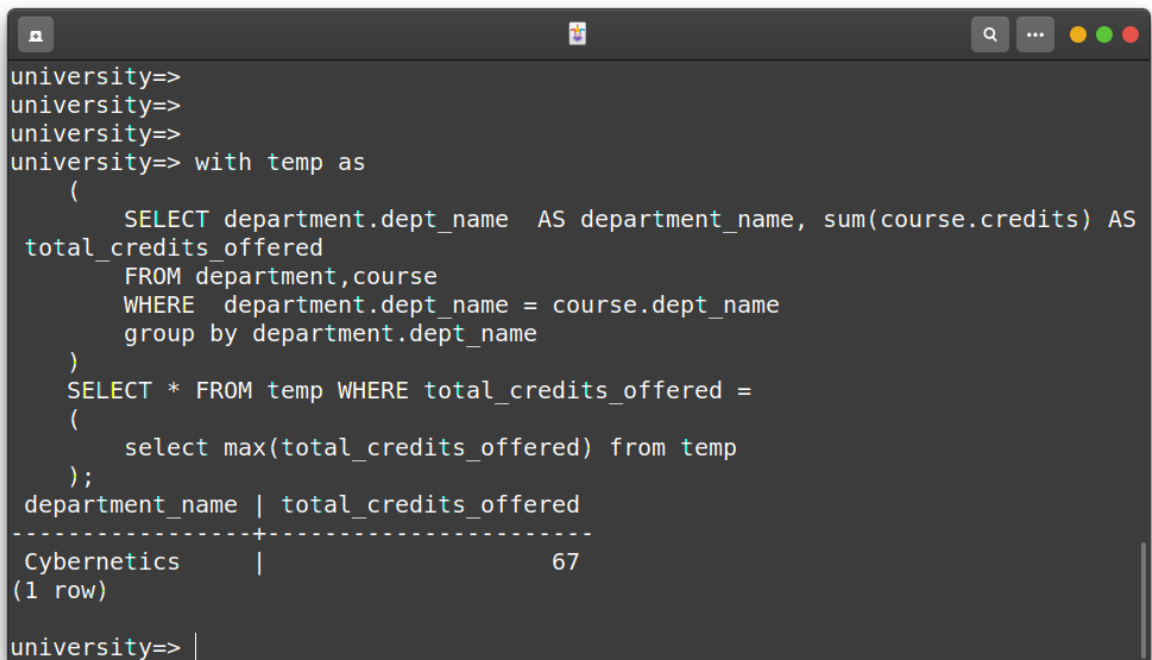
```
university=>
university=>
university=>
university=>
university=> SELECT * FROM TotalCredits
university-> WHERE total_credits_offered = (select max(total_credits_offere
d) from TotalCredits);
  department_name | total_credits_offered
-----+-----
Cybernetics      | 67
(1 row)

university=>
university=>
university=> |
```

Base Table Query:

with temp as

```
(
  SELECT department.dept_name AS department_name,
  sum(course.credits) AS total_credits_offered
  FROM department,course
  WHERE department.dept_name = course.dept_name
  group by department.dept_name
)
SELECT * FROM temp WHERE total_credits_offered =
(
  select max(total_credits_offered) from temp
);
```



The screenshot shows a terminal window with a dark background. The prompt is 'university=>'. The user has entered a SQL query that uses a CTE named 'temp' to find the department with the highest total credits offered. The query is as follows:

```
university=>
university=>
university=>
university=> with temp as
(
  SELECT department.dept_name AS department_name, sum(course.credits) AS
total_credits_offered
  FROM department,course
  WHERE department.dept_name = course.dept_name
  group by department.dept_name
)
SELECT * FROM temp WHERE total_credits_offered =
(
  select max(total_credits_offered) from temp
);
```

The result of the query is displayed as a table with two columns: 'department_name' and 'total_credits_offered'. The table has one row with the values 'Cybernetics' and '67'. Below the table, it says '(1 row)'. The prompt 'university=>' is followed by a vertical bar '|', indicating the cursor is at the end of the line.

department_name	total_credits_offered
Cybernetics	67

(1 row)

university=> |