**CS1555 Recitation 7 Solution**

Objective:

1. To practice more SQL queries on PostgreSQL.
2. To practice Views

**PART 1:**

Before we start:

* Download the SQL script studentdb.sql through an sFTP client (such as FileZilla) from the machine “class3.cs.pitt.edu” at the directory:
  + /afs/pitt.edu/home/r/a/raa88/public/studentdb.sql

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. For each course a student from ‘CS’ major has repeated, list the student id and course number.

SELECT S.sid, CT.course\_no, COUNT(\*)

FROM COURSE\_TAKEN CT JOIN STUDENT S on CT.sid = S.sid

WHERE major = ‘CS’

GROUP BY S.sid, CT.course\_no

HAVING COUNT(\*) > 1 ;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. List the sid(s) and names of the students who have not taken the course “Web Applications”.

--Solution 1: using set difference (notice no one appears, add grade is not null inside)

SELECT sid, name

FROM STUDENT

WHERE sid NOT IN (

SELECT sid

FROM COURSE\_TAKEN CT, COURSE C

WHERE CT.course\_no = C.course\_no

AND C.name = ‘Web Applications’);

--Solution 2: equivalently, you can use the "exists" operator as follows:

SELECT S.sid, S.name

FROM STUDENT S

WHERE NOT EXIST (

SELECT \*

FROM COURSE\_TAKEN CT, COURSE C

WHERE CT.course\_no = C.course\_no

AND C.name = 'Web Applications’

AND CT.sid = S.sid);

--Solution 3: using outer join

SELECT S.sid, S.name

FROM STUDENTS S LEFT OUTER JOIN (

SELECT sid, course\_no

FOM COURSE\_TAKEN CT NATURAL JOIN COURSE

WHERE name = 'Web Applications') WA\_TAKING

ON S.sid = WA\_TAKING.sid

WHERE course\_no IS NULL;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Find the top 3 students with the highest GPAs.

--note that if all the grades of a student is null, the average (GPA) will be null. Ordering by GPA, those with null GPA will appear first. Therefore, we specify a condition "avg(grade) is not null" in order to eliminate those tuples with null GPA to appear in the result set.

SELECT \*

FROM (

SELECT sid, AVG(grade) AS GPA

FROM COURSE\_TAKEN

GROUP BY sid

HAVING AVG(grade) IS NOT NULL

ORDER BY AVG (grade) DESC

) AS TOP3

FETCH FIRST 3 ROWS ONLY;

Is there another way to do that? (exercise)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Find the sid and GPA of the top 1 student whose GPA is greater than the student whose sid is 123.

SELECT ct1.sid, avg(ct1.grade) as GPA

FROM COURSE\_TAKEN ct1

GROUP BY ct1.sid

HAVING AVG (CT1.grade) > (

SELECT AVG(CT2.grade) FROM COURSE\_TAKEN CT2 WHERE CT2.sid = '123')

ORDER BY AVG(ct1.grade) DESC

FETCH FIRST 1 ROWS ONLY;

5. Create a view called STUDENT\_COURSES that lists the sid(s), student names, number of courses in the COURSE\_TAKEN table.

CREATE OR REPLACE VIEW STUDENT\_COURSES AS

SELECT S.sid, S.name, count(course\_no) AS num\_courses

FROM STUDENT S, COURSE\_TAKEN CT

WHERE S.sid = CT.sid

GROUP BY S.sid, S.name;

6. Rank the students (sid and name) based on their GPA. Can we do something simpler?

|  |  |
| --- | --- |
| **select sid**, **name**,  (1 + (**select** *count*(*\**)  **from** (**select s**.**sid**, **s**.**name**, *avg*(**grade**) **as** gpa  **from** COURSE\_TAKEN ct  **join** student **s on** ct.**sid** = **s**.**sid  where grade is not null  group by s**.**sid**, **s**.**name  having** *avg*(**grade**) > i.gpa  **order by** gpa)e)  ) **as** rank **from** (**select s**.**sid**, **s**.**name**, *avg*(**grade**) **as** gpa  **from** COURSE\_TAKEN ct  **join** student **s on** ct.**sid** = **s**.**sid  where grade is not null  group by s**.**sid**, **s**.**name  order by** gpa)i **order by** rank; | *-- Simplify* **create or replace view** student\_gpa **as select s**.**sid**, **s**.**name**, *avg*(**grade**) **as** gpa **from** COURSE\_TAKEN ct  **join** student **s on** ct.**sid** = **s**.**sid where grade is not null group by s**.**sid**, **s**.**name order by** gpa;  *-- Now the query* **select i.sid**, i.**name**,  (1 + (**select** *count*(*\**)  **from** student\_gpae  **where** e.**gpa** > i.**gpa**)  ) **as** rank **from** student\_gpai **order by** rank; |
| Is there another way to do that?  *-- Using Rank()*  **SELECT sid, name, RANK() OVER (**  **ORDER BY** gpa **DESC**  **) AS** rank  **FROM STUDENT\_GPA** |  |