**CS1555 Recitation 6 Solution**

**Objective:** To practice SQL queries on PostgreSQL.

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. Find the address and phone number of the student whose SID is 123

**select** address, phone

**from** student\_dir

**where** sid = 123**;**

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

1. List all the courses offered in ‘Spring 11’.

**select** **distinct** course\_no

**from** course\_taken

**where** term = 'Spring 11'**;**

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

3. List the student ID and course number for every student who took a course in Fall 10 but has not received a grade yet.

**select** ct.sid, ct.course\_no

**from** course\_taken ct

**where** ct.term = 'Fall 10'

**and** ct.grade **is null;**

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

4. List the SIDs *and GPAs* of the students whose GPAs are greater than 3.7. List them in the descending order of the GPAs.

**select** SID, **avg**(grade) as GPA

**from** course\_taken

**group by** sid

**having avg**(grade) > 3.7

**order by avg**(grade) **desc;**

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

5. List the SIDs of all the students and the number of courses they have taken.

**select** sid, **count(distinct** course\_no) **as** num\_courses

**from** course\_taken

**group by** sid;

What if we want names too?

**select** s.sid, s.name, **count(distinct** course\_no) **as** num\_courses

**from** student s **join** course\_taken ct **on** s.sid = ct.sid

**group by** s.sid, s.name;

How about another way?

**select** sid, name, **count(distinct** course\_no) **as** num\_courses

**from** student s **natural join** course\_taken ct

**group by** sid, name;

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

6. Now insert a tuple into the Student table:

insert into student values (130, 'Peter', 1,'CS', '????');

Then run the query 5 again. How can we include this new student in the result, with 0 as the number of classes he has taken?

**select** s.sid, s.name, **count**(distinct course\_no) **as** num\_courses

**from** student s **left outer join** course\_taken ct **on** s.sid = ct.sid

**group by** s.sid, s.name;

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

7. For each course a student has repeated, list the studentID and course number.

**select** sid, course\_no, **count**(\*)

**from** course\_taken

**group by** sid, course\_no

**having** count(\*) >1 ;