Ace Cassidy CPTS451 HW4 Spring'20

1. Find the distinct courses that "Software Engineering (SE)" track students in 'CptS'major are enrolled in. Return the major, courseNo, and credits for those courses –sorted by major and courseNo.

```
SELECT distinct enroll.coursemajor, enroll.courseno, credits
FROM Student, Course, Enroll
WHERE studentmajor='CptS' AND trackcode='SE'
    AND enroll.sID = student.sID
    AND course.courseno = enroll.courseno
    AND course.courseMajor = enroll.courseMajor
ORDER BY enroll.coursemajor, enroll.courseno;
```

2. Find the sorted names, ids, majors and track codes of the students who are enrolled in more than 18 credits (19 or above).

```
SELECT sname, student.sID, studentmajor, trackcode, SUM(credits)
FROM Student, Course, Enroll
WHERE enroll.sID = student.sID AND course.courseno = enroll.courseno
    AND course.courseMajor = enroll.courseMajor
GROUP BY student.sID
HAVING SUM(credits) >= 18
ORDER BY sname;
```

3. Find the courses that only 'SE' track students in 'CptS major have been enrolled in. Give an answer without using the set EXCEPT operator.

```
SELECT distinct enroll.coursemajor, enroll.courseno
FROM Student, Course, Enroll
WHERE (enroll.coursemajor, enroll.courseno) NOT IN (
    SELECT enroll.coursemajor, enroll.courseno
    FROM Student, Course, Enroll
    WHERE   enroll.sID = student.sID
        AND enroll.courseno = course.courseno
        AND enroll.courseMajor = course.courseMajor
        AND (studentmajor<>'CptS' OR trackcode<>'SE')
    ) AND studentmajor IS NOT NULL and trackcode IS NOT NULL;
```

4. Find the students who were enrolled in the courses that 'Ali' was enrolled in and earned the same grade as 'Ali' in those courses. Return the name, id, and major of the student as well as the courseNo, major, and student's grade for those courses.

```
SELECT s2.sname, s2.sID, s2.studentmajor,
    e2.coursemajor, e2.courseno, e2.grade
FROM Student as s1, enroll as e1, student as s2, enroll as e2
WHERE   s1.sname='Ali'
    AND s1.sid = e1.sid
    AND s2.sid = e2.sid
    AND e2.grade = e1.grade
    AND e2.courseno = e1.courseno
    AND e2.coursemajor = e1.coursemajor
    AND s2.sname<>'Ali';
```

5. Find the students in 'CptS' major who are not enrolled in any classes. Return their names and sIDs. (Note: Give a solution using OUTER JOIN)

```
SELECT sname, student.sid
FROM student
FULL OUTER JOIN enroll
ON student.sid=enroll.sid
WHERE   student.sid IS NOT NULL
```

```
    AND student.studentmajor='CptS';
    AND enroll.courseno IS NULL
```

6. Find the courses whose enrollments exceed their enrollment limits (i.e., the total enrollment is greater than the enrollment limit of the course). Return the course major, courseNo, enrollment limit, and the actual enrollment for those courses.

```
SELECT enroll.coursemajor, enroll.courseno, enroll_limit, COUNT(enroll.sid) as enrollnum
FROM   course, enroll
WHERE   course.coursemajor=enroll.coursemajor
    AND course.courseno=enroll.courseno
GROUP BY (enroll.coursemajor, enroll.courseno, course.enroll_limit )
HAVING COUNT(enroll.sid) > enroll_limit;
```

7. Find the courses which are prerequisites for more than 5 courses. Return the major, courseNo, and the number of the successor courses.

```
SELECT pre.coursemajor, pre.courseno, COUNT(course.precourseno)
FROM prereq as pre, prereq as course
WHERE    course.premajor = pre.coursemajor
    AND course.precourseno = pre.courseno
GROUP BY (pre.coursemajor, pre.courseno)
HAVING COUNT(course.precourseno) > 5;
```

8. Find the 'CptS' major students who passed a course but failed the prerequisite of that course, i.e., got a grade lower than "2". (For example, Alice (sid: 12583589) passed CptS355 but had a grade 1.75 in the prerequisite course CptS223.) Return the names and sIDs of those students and the courseno of the course (i.e., the course whose prereq had a low grade).

```
SELECT student.sname, student.sid, enroll.coursemajor, enroll.courseno
FROM student, enroll, prereq, enroll as preenroll
WHERE    student.studentmajor='CptS'
    AND enroll.sid = student.sid
    AND enroll.coursemajor = prereq.coursemajor
    AND enroll.courseno = prereq.courseno
    AND enroll.grade >= 2
    AND preenroll.sid = student.sid
    AND preenroll.coursemajor = prereq.premajor
    AND preenroll.courseno = prereq.precourseno
    AND preenroll.grade < 2;
```

9. For each 'CptS' course, find the percentage of the students who passed the course. Assume a passing grade is 2 or above. (Note: Assume that students who didn't earn a grade in class should be excluded in average calculation).

```
SELECT enroll.coursemajor, enroll.courseno,
    round( avg
        (case when grade > 2 then 100.0 else 0.0 end)
    ,0) as passrate
FROM student, enroll
WHERE   enroll.sid = student.sid
    AND enroll.coursemajor = 'CptS'
    AND enroll.grade IS NOT NULL
GROUP BY (enroll.coursemajor, enroll.courseno)
ORDER BY enroll.courseno;
```

10. Write the equivalent SQL query for the following relational algebra expressions.

- R1= Student (Student.sID=Enroll.sID AND Student.studentMajor=Enroll.courseMajor)Enroll
- R2= gpa>=2(sID,avg(grade)->gpaR1)) Student Student.studentMajor=Majors.majorMajors

- sID,sName,studentMajor,description,gpa R2

```
SELECT student.sid, sname, studentmajor, description, gpa
FROM student, majors,
    (SELECT enroll.sid , avg(enroll.grade) as gpa
    FROM student, enroll
    WHERE enroll.sid = student.sid
        AND enroll.coursemajor = student.studentmajor
    GROUP BY enroll.sid
    HAVING avg(enroll.grade) >= 2
    ) as passing
WHERE student.studentmajor = majors.major
    AND student.sid=passing.sid
ORDER BY student.sid;
```