Consider the following relations:

- **Course(**cid, deptName, cName, year, majorRequired)
    - The primary key of this relation is {cid, deptName} (e.g., 304, CPSC)
    - cid is a course id, e.g., 110, 304;
    - deptName is the name of the department, e.g., CPSC, MATH;
    - cName is the name of the course, e.g., “database management”;
    - year is an integer in the set {1, 2, 3, 4};
    - majorRequired is a Boolean value which is either true or false.

- **Prereq(**cid, deptName, prereqCid, prereqDeptName)
    - The primary key for this relation is {cid, deptName, prereqCid, prereqDeptName }
    - This is a relation that stores the prerequisite courses for the course (cid, deptName);
    - Each prerequisite course is denoted by (prereqCid, prereqDeptName).

- **Offering(**cid, deptName, calendarYear, term, enrollment, avgGrade)
    - The primary key for this relation is {cid, deptName, calendarYear, term}
    - This is a relation that stores all the offerings of a course (cid, deptName);
    - calendarYear is the year the course was offered, e.g., 2017, 2018;
    - term is an integer in the set {1, 2, 3}, i.e., there are only 3 terms per year;
    - enrollment is the number of students enrolled in that course during that year and that term;
    - avgGrade is the average grade of the class for that offering; it ranges from 0% to 100%.

- **Took(**sid, cid, deptName, calendarYear, term, grade)
    - The primary key of this relation is {sid, cid, deptName, calendarYear, term }
    - This is a relation that records the grade of a student taking a particular course during a particular year and term;
    - sid is the student id of the student;
    - (cid, deptName, calendarYear, term) specifies a particular offering of a course;
    - grade is the grade of the student who took that offering of the course; it ranges from 0% to 100%, without any null values. The passing grade of any course is 50%.

Write an SQL query to answer each of the following questions.

1. Find the student id who has taken the most courses (i.e., which student id has taken the highest number of courses)?
2. For each course, sum up the number of students that have taken that specific course.
3. Find all students who has taken all CPSC classes
4. Oops! The university ran out of money. Delete all the courses that are not required for a major.
5. Oops! The university made a mistake tabulating grades. For every course offered in 2007, update the course average to be 10% higher than what is currently listed.
6. Find the course with the highest number of offerings.
7. Return all the courses that have the word “uwu” in their course names.
8. Find the course that has the most number of prerequisites.
9. Find all the courses taken by sid = 1010102 where the final course grade was a B (for the purposes of the problem, a B is 72-80).

Answers:

1.
```
SELECT sid
FROM Took
group by sid
HAVING count(cid) = (SELECT MAX(countCourses)
                      FROM (SELECT count(*) AS countCourses
                              FROM Took
                              GROUP BY sid)
                      )
```

2.
```
SELECT deptName, cid, COUNT(*) AS studentsTakenTotal
FROM Took
GROUP BY deptName, cid
```

3.
```
SELECT s.sid
FROM Students s
WHERE NOT EXISTS (
        (SELECT c.cid FROM Course c WHERE c.deptName = "CPSC")
        EXCEPT
        (SELECT t.cid FROM Took t WHERE t.sid = s.sid )

)
```

4.
```
DELETE FROM Course
WHERE majorRequired = false;
```

   Note: side effects may occur in PreReq and Offering so you may have other tuples that also need to be deleted as well

5.
```
UPDATE offering
SET avgGrade = avgGrade + 10
WHERE calendarYear – 2007
```

6.

```
CREATE VIEW numCourseOfferings(deptName, cid, numOfferings) AS
        SELECT deptName, cid, COUNT(*) AS numOfferings
        FROM Offering
        GROUP BY deptName, cid

SELECT deptName, cid
FROM numCourseOfferings n1
WHERE n1.numOfferings >= ALL (SELECT n2.numOfferings
                               FROM numCourseOfferings n2)
```

7.

```
SELECT deptName, cid
FROM courses
WHERE cName LIKE '%uwu%';
```

8.

```
CREATE VIEW numPrereq(deptName, cid, numPrereq) AS
        SELECT deptName, cid, count(*) AS numPrereq
        FROM Prereq p
        GROUP BY deptName, cid

SELECT deptName, cid
FROM  numPrereq
WHERE numPreq = (SELECT MAX(numPrereq) FROM numPrereq)
```

9.

```
SELECT DISTINCT deptName, cid
FROM Took
WHERE sid = 1010102 AND grade > 72 AND grade < 80
```