1. Create the database schema

```
CREATE TABLE students (
sid VARCHAR(20),
name VARCHAR(50),
age INTEGER,
gpa FLOAT,
PRIMARY KEY(sid)
);

CREATE TABLE courses (
cid VARCHAR(20),
deptid VARCHAR(20),
name VARCHAR(50),
PRIMARY KEY(cid)
);

CREATE TABLE professors (
SSN INTEGER,
name VARCHAR(50),
address VARCHAR(30),
phone VARCHAR(10),
deptid VARCHAR(20),
PRIMARY KEY(SSN)
);

CREATE TABLE enrollment (
sid VARCHAR(20),
cid VARCHAR(20),
section INTEGER,
grade VARCHAR(2),
PRIMARY KEY(sid, cid),
FOREIGN KEY(sid) REFERENCES students,
FOREIGN KEY(cid) REFERENCES courses,
FOREIGN KEY(cid, section) REFERENCES teaches
);
```

```
CREATE TABLE teaches (
cid VARCHAR(20),
section INTEGER,
ssn INTEGER,
PRIMARY KEY (cid, section),
FOREIGN KEY (cid) REFERENCES courses,
FOREIGN KEY (ssn) REFERENCES professors
);
```

2. Find CS professor names

```
SELECT name FROM professors WHERE deptid = 'CS';
```

3. Find those students (sid) enrolled in courses in the CS department

```
SELECT S.sid FROM students AS s INNER JOIN enrollment AS e
ON s.sid = e.sid INNER JOIN courses AS c ON e.cid = c.cid
WHERE c.deptid = 'CS';
```

4. List ssn & name of professors in CS department teaching no classes

```
SELECT p.ssn, p.name FROM professors AS p JOIN teaches AS t
ON p.ssn = t.ssn JOIN courses AS c ON c.cid = t.cid
WHERE p.deptid = 'CS' and c.deptid <> 'CS';
```

5. List only the number of courses offered by each department

```
SELECT deptid, COUNT(cid) FROM courses GROUP BY deptid;
```

6. List departments that offer more than 10 courses

```
SELECT deptid, COUNT(cid) FROM courses GROUP BY deptid HAVING COUNT(cid)>10;
```

7. List students whose professors names start with 'M' without duplicates

```
SELECT DISTINCT S.name FROM students AS s INNER JOIN enrollment AS e
ON e.sid = s.sid INNER JOIN teaches AS t ON t.cid = e.cid INNER JOIN
professors AS p ON p.ssn = t.ssn WHERE p.name LIKE 'M%';
```

8. small sections < 30, 80 > medium sections ≥ 30, large sections > 80

```
SELECT deptid, count(sid) < 30 AS 'Small section',
count(sid) >= 30 and count(sid) < 80 AS 'Medium Section',
count(section) >= 80 AS 'large section' from enrollment E,
courses C where E.cid = C.cid GROUP BY E.section, C.cid;
```

9. List professors in departments > 20 faculty members that offer more large sections than small and medium sections combined

```
select p.name as professors from professors p, courses c, teaches t
where p.deptid = c.deptid group by p.name having count(p.name) > 20;
```

10. Assume possible grades are A, B, C, D, F, where D and F are failing grades. Find the percentage of students failing each course.

```
Select t1.section, t1.grade, (count(t1.grade)* 100 /(select count (*)
From enrollment where t1.grade in ('D', 'F')))
'score' from enrollment t1, courses t2 where t1.cid = t2.cid
group by t1.section, t1.grade;
```

11. Find the name of the professor with the maximum percentage of students that failed the course

```
Select distinct name from professors p, enrollment e where grade in('F')
group by name;
```

12. On average, what percentage of students fail a course

```
Select 100*(select count (gpa) FROM students WHERE gpa<3.3)/
(count(gpa)) "% gpa"
FROM students;
```

13. Find list of courses (sections) where the percentage of students with D or F is greater than average

```
Select section from enrollment e, courses c, where grade in ('F', 'D')
group by section having count (grade) > (select avg (section) from enrollment);
```

14. Query to produce the following table

```
SELECT c.deptid AS deptid,
ROUND (AVG (enrollment.section_students), 2) AS SPS,
ROUND (100 * SUM (CASE WHEN enrollment.grade = 'A' THEN 1 ELSE 0 END)
        COUNT(*), 2) AS '%A',
ROUND (100 * SUM (CASE WHEN enrollment.grade = 'B' THEN 1 ELSE 0 END)
        COUNT(*), 2) AS '%B',
ROUND (100 * SUM (CASE WHEN enrollment.grade = 'C' THEN 1 ELSE 0 END)
        COUNT(*), 2) AS '%C',
```

```
ROUND(100 * SUM(CASE WHEN enrollment.grade = 'D' THEN 1 ELSE 0 END)
      COUNT(*), 2) AS '%D',
ROUND(100 * SUM(CASE WHEN enrollment.grade = 'F' THEN 1 ELSE 0 END)
      COUNT(*), 2) AS '%F',

FROM enrollment,
JOIN courses C ON enrollment.cid = c.cid,
GROUP BY c.deptid,
ORDER BY c.deptid;
```