You implement question 1 and 2 in MySQL. Show with actual tables and MySQL prompts that your SQLs are working. Question 3 does not have to be implemented.

1. Predicate Logic:
   
   Query language for relational model was first proposed based on predicate logic. Consider the following predicate logic statement.
   
   \[ \{ y \mid \forall_x (S(y) \land CD(x,"CS") \rightarrow SC(x, y)) \} \]
   
   Where Variable x: Universe of discourse is all courses in the University.
   
   CD(x,z): course x is offered by department z
   
   SC(x, y): student y is taking course x, y is a free variable
   
   S(y): y is a student, Universe of discourse for y is all people.
   
   (a) (5 points) Express the query implemented by the above predicate logic statement in plain English (without using the terms quantifier, implications, etc).
   
   (b) (10 points) Implement the above predicate logic statement into SQL. Because SQL does not provide universal quantifier, we convert the above into a statement using only existential quantifiers as follows:
   
   \[ y \mid S(y) \land \forall_x (CD(x,"CS") \rightarrow SC(x, y)) \]
   
   Logical equivalence of: \( p \rightarrow q \) and \( \neg p \lor q \)
   
   \[ y \mid S(y) \land \forall_x (\neg CD(x,"CS") \lor SC(x, y)) \]
   
   Logical equivalence of: \( \forall_x p(x) \) and \( \neg \exists_x \neg p(x) \)
   
   \[ y \mid S(y) \land \neg \exists_x ((\neg CD(x,"CS") \lor SC(x, y))) \]
   
   De Morgan’s Laws
   
   \[ y \mid S(y) \land \neg \exists_x CD(x,"CS") \land \neg SC(x, y) \]
   
   \( \neg SC(x, y) \) implemented in SQL by \( \neg \exists_z (SC(z, y) \land z \neq x) \)
   
   \[ y \mid S(y) \land \neg \exists_x CD(x,"CS") \land \neg \exists_z (SC(z, y) \land z = x) \]
   
   Implement the above transformed predicate logic statement into Mysql and show which parts of the predicate logic statement correspond to which parts of the SQL statement. You create table for students \( S(S_i,d) \), table for student course \( SC(S_i,d,C_i,d) \) and a table for course department \( CD(C_i,d,D_i,d) \)
2. Create a new table Department: D(d-id, s-id) and extend the SC table to add the grade attribute: SC(s-id, C-id, grade). Implement SQL statements for the following and run it on Mysql.

(a) (10 points) Find those departments whose average grade in all courses offered by the department is greater than 2.0 and the number of students in the department is greater than 100.

(b) (15 points) Find those departments who have not offered any courses with minimum average grade in a course less than 3.0 and have at least two students in the department with GPA 4.0

NAVIGATIONAL QUERIES (navigating through various tables using foreign key referencing).

3. (10 points) Based on the foreign key reference graph (posted separately) answer the following query:

Give the instructors of those students, the corresponding Departments the instructors belong to and the departments for those students, who are in degree program with program ID= P100.

In navigational queries, semantic depends on which path of the foreign key reference graph you chose. Some times there may be more than one path and you have to select the one that is most appropriate for your query. Make your own assumptions here, if you need to.

EXAMPLE:

Give all students, their degree programs and their courses.

Select *
From Student s, DegreeProgram d, taken t, CourseOffering o, CourseDescription c
Where s. Major= d.ProId and s.Sid=t.Sid and t.SeqId=o.SeqId and o.Cno=c.Cno