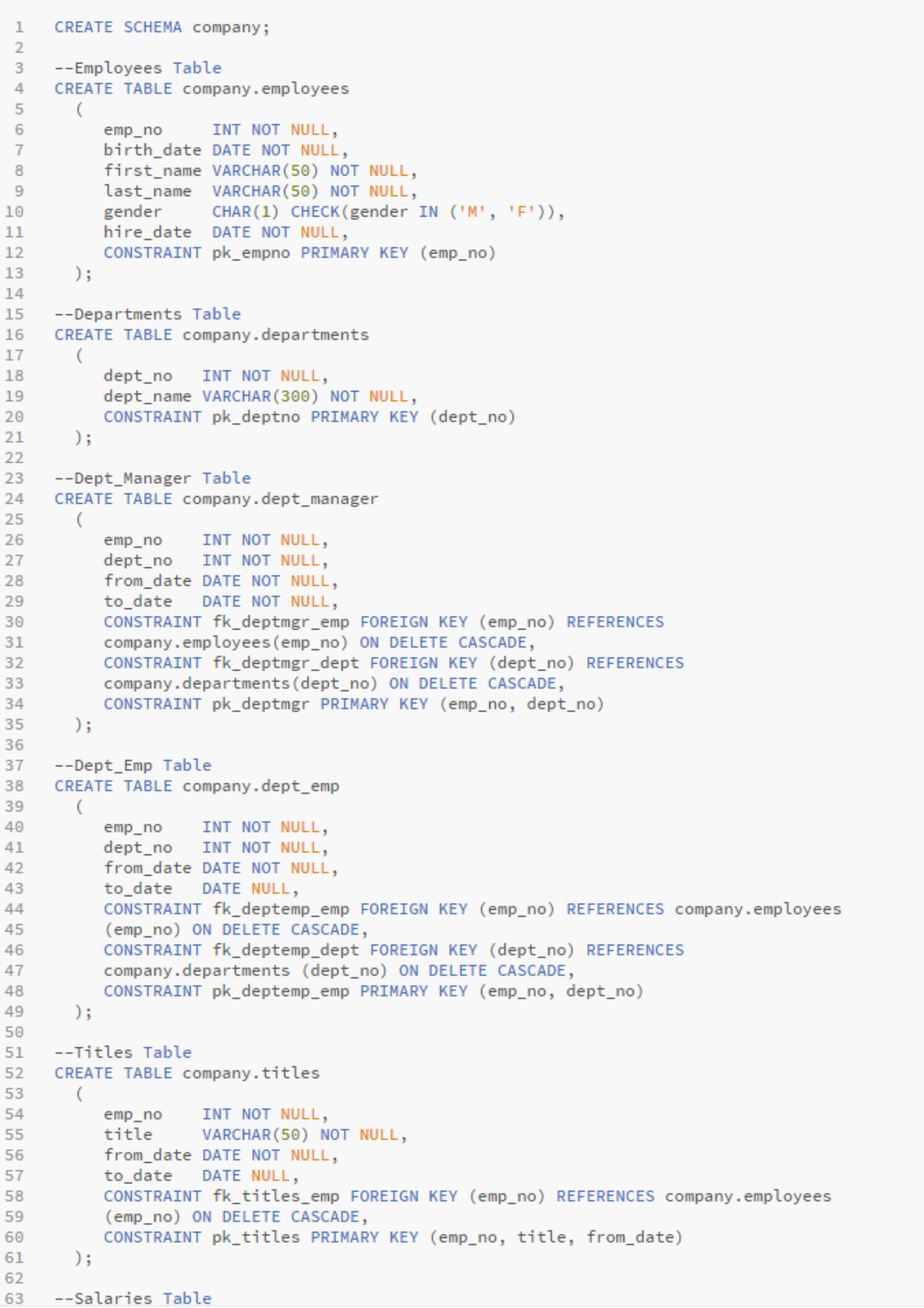
HOMEWORK 1 – RISHIRAJ GOHIL

QUESTION 1

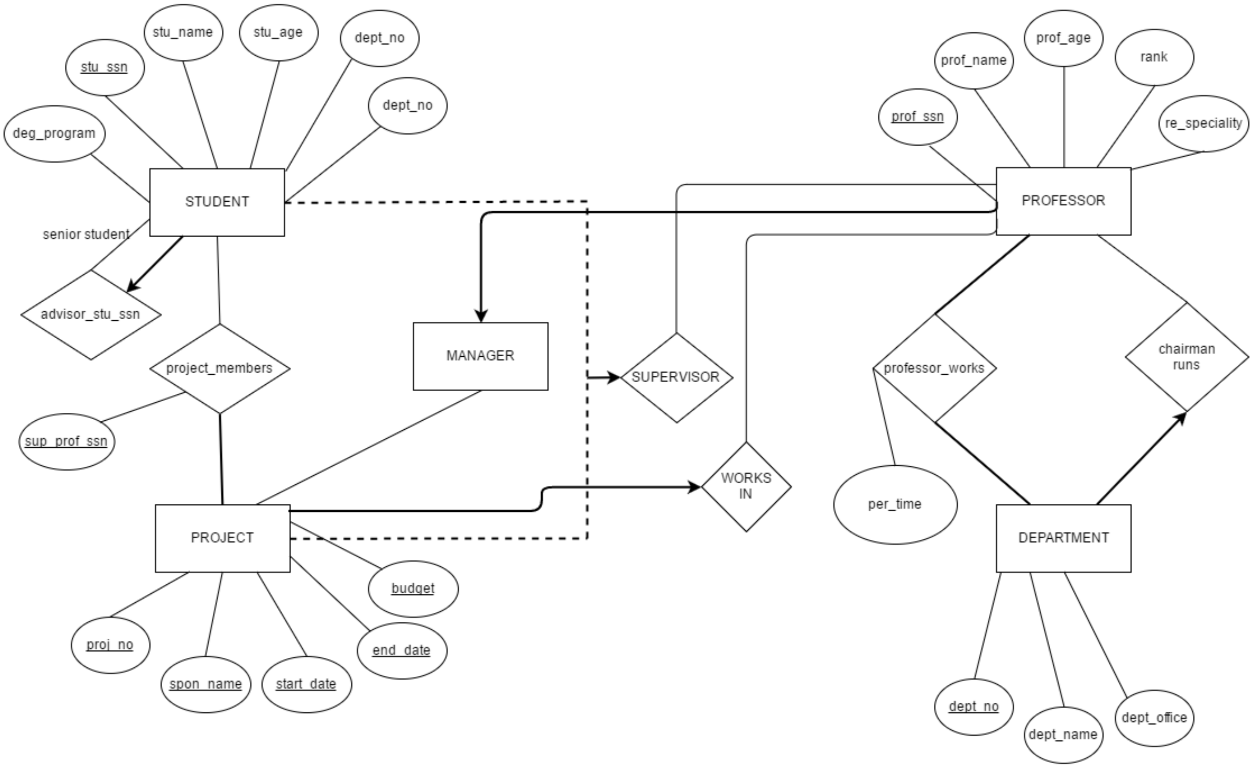
Answers in **Q1_Company_Schema.sql**



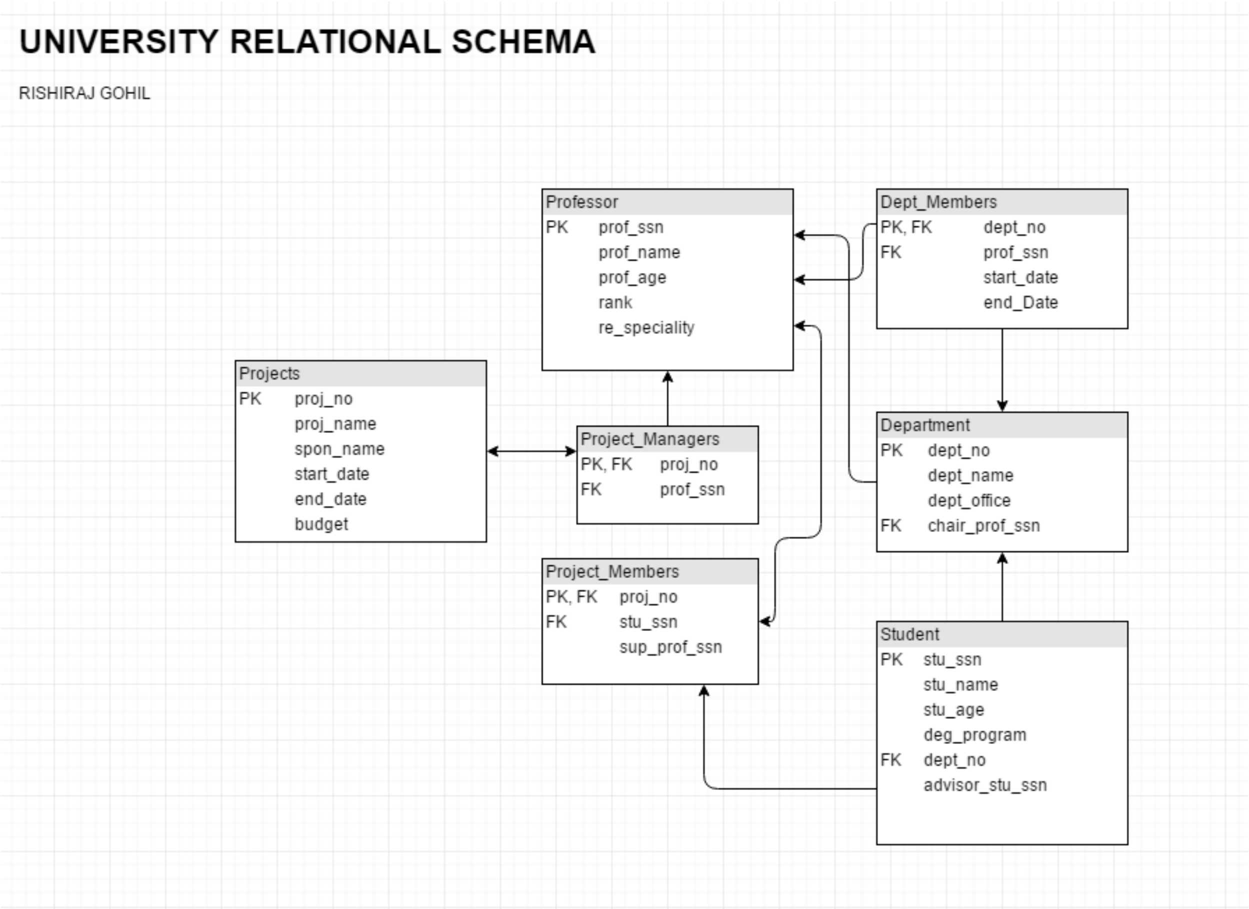
```
CREATE SCHEMA company;

--Employees Table
CREATE TABLE company.employees
  (
     emp_no     INT NOT NULL,
     birth_date DATE NOT NULL,
     first_name VARCHAR(50) NOT NULL,
     last_name  VARCHAR(50) NOT NULL,
     gender     CHAR(1) CHECK(gender IN ('M', 'F')),
     hire_date  DATE NOT NULL,
     CONSTRAINT pk_empno PRIMARY KEY (emp_no)
  );

--Departments Table
CREATE TABLE company.departments
  (
     dept_no   INT NOT NULL,
     dept_name VARCHAR(300) NOT NULL,
     CONSTRAINT pk_deptno PRIMARY KEY (dept_no)
  );

--Dept_Manager Table
CREATE TABLE company.dept_manager
  (
     emp_no    INT NOT NULL,
     dept_no   INT NOT NULL,
     from_date DATE NOT NULL,
     to_date   DATE NOT NULL,
     CONSTRAINT fk_deptmgr_emp FOREIGN KEY (emp_no) REFERENCES
     company.employees(emp_no) ON DELETE CASCADE,
     CONSTRAINT fk_deptmgr_dept FOREIGN KEY (dept_no) REFERENCES
     company.departments(dept_no) ON DELETE CASCADE,
     CONSTRAINT pk_deptmgr PRIMARY KEY (emp_no, dept_no)
  );

--Dept_Emp Table
CREATE TABLE company.dept_emp
  (
     emp_no    INT NOT NULL,
     dept_no   INT NOT NULL,
     from_date DATE NOT NULL,
     to_date   DATE NULL,
     CONSTRAINT fk_deptemp_emp FOREIGN KEY (emp_no) REFERENCES company.employees
     (emp_no) ON DELETE CASCADE,
     CONSTRAINT fk_deptemp_dept FOREIGN KEY (dept_no) REFERENCES
     company.departments (dept_no) ON DELETE CASCADE,
     CONSTRAINT pk_deptemp_emp PRIMARY KEY (emp_no, dept_no)
  );

--Titles Table
CREATE TABLE company.titles
  (
     emp_no    INT NOT NULL,
     title     VARCHAR(50) NOT NULL,
     from_date DATE NOT NULL,
     to_date   DATE NULL,
     CONSTRAINT fk_titles_emp FOREIGN KEY (emp_no) REFERENCES company.employees
     (emp_no) ON DELETE CASCADE,
     CONSTRAINT pk_titles PRIMARY KEY (emp_no, title, from_date)
  );

--Salaries Table
```

HOMEWORK 1 – RISHIRAJ GOHIL

Question 2



Question 3



UNIVERSITY RELATIONAL SCHEMA

RISHIRAJ GOHIL

Professor	
PK	prof_ssn
	prof_name
	prof_age
	rank
	re_speciality

Dept_Members	
PK, FK	dept_no
FK	prof_ssn
	start_date
	end_Date

Projects	
PK	proj_no
	proj_name
	spon_name
	start_date
	end_date
	budget

Project_Managers	
PK, FK	proj_no
FK	prof_ssn

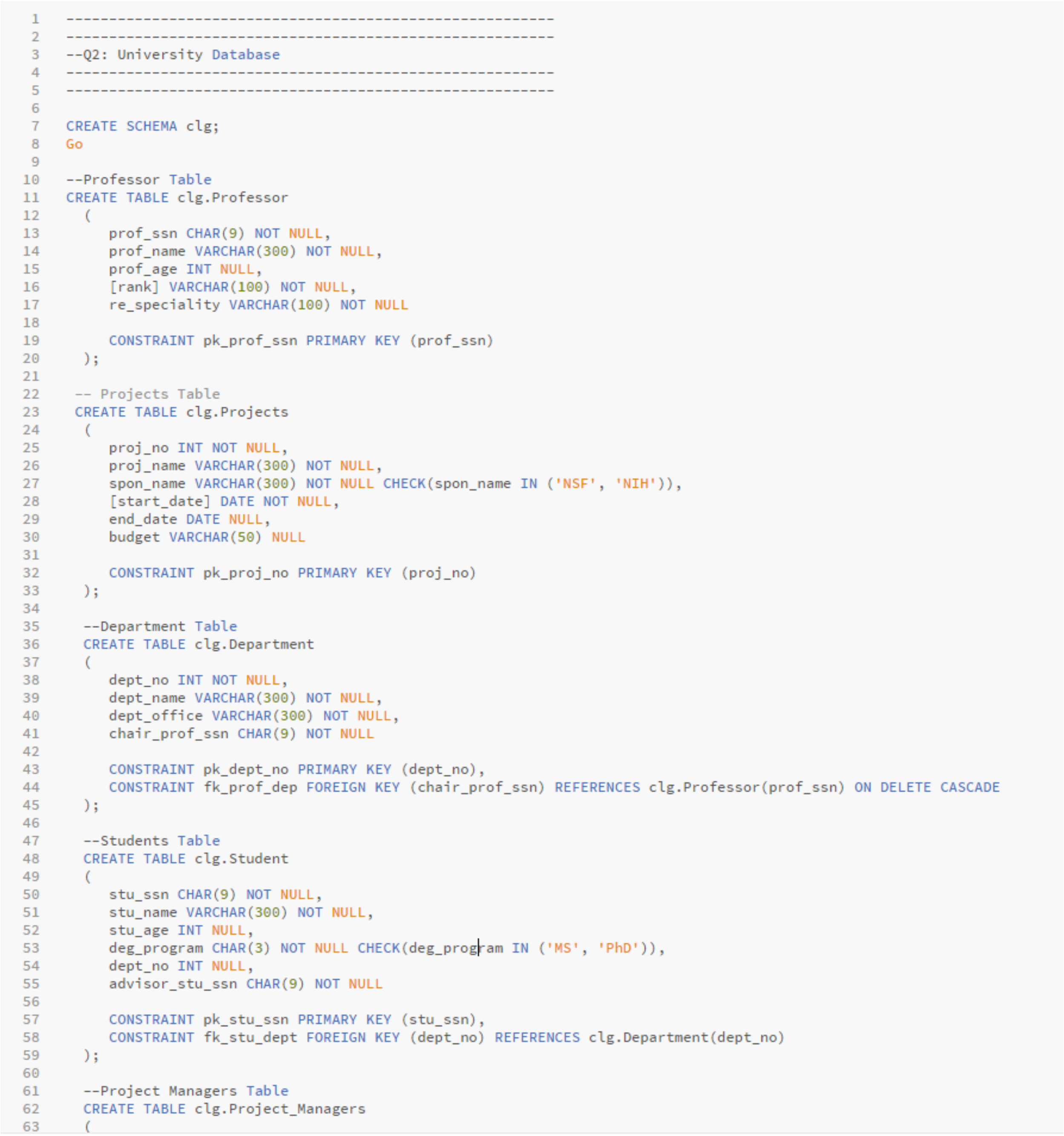
Department	
PK	dept_no
	dept_name
	dept_office
FK	chair_prof_ssn

Project_Members	
PK, FK	proj_no
FK	stu_ssn
	sup_prof_ssn

Student	
PK	stu_ssn
	stu_name
	stu_age
	deg_program
FK	dept_no
	advisor_stu_ssn

Question 4

SQL in File: **Q2_University_Database_Schema-postgre.sql**



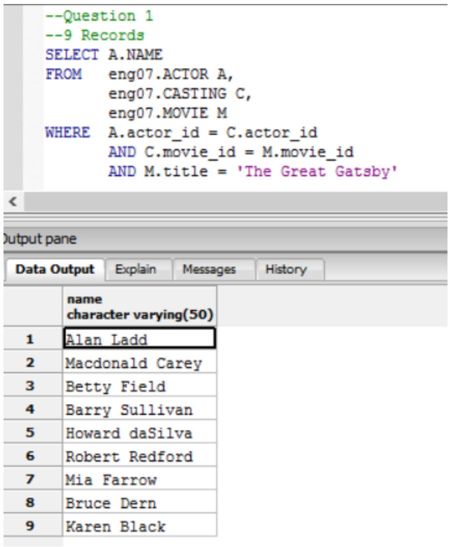
```
---------------------------------------------------------
---------------------------------------------------------
--Q2: University Database
---------------------------------------------------------
---------------------------------------------------------

CREATE SCHEMA clg;
Go

--Professor Table
CREATE TABLE clg.Professor
  (
     prof_ssn CHAR(9) NOT NULL,
     prof_name VARCHAR(300) NOT NULL,
     prof_age INT NULL,
     [rank] VARCHAR(100) NOT NULL,
     re_speciality VARCHAR(100) NOT NULL

     CONSTRAINT pk_prof_ssn PRIMARY KEY (prof_ssn)
  );

 -- Projects Table
 CREATE TABLE clg.Projects
  (
     proj_no INT NOT NULL,
     proj_name VARCHAR(300) NOT NULL,
     spon_name VARCHAR(300) NOT NULL CHECK(spon_name IN ('NSF', 'NIH')),
     [start_date] DATE NOT NULL,
     end_date DATE NULL,
     budget VARCHAR(50) NULL

     CONSTRAINT pk_proj_no PRIMARY KEY (proj_no)
  );

  --Department Table
  CREATE TABLE clg.Department
  (
     dept_no INT NOT NULL,
     dept_name VARCHAR(300) NOT NULL,
     dept_office VARCHAR(300) NOT NULL,
     chair_prof_ssn CHAR(9) NOT NULL

     CONSTRAINT pk_dept_no PRIMARY KEY (dept_no),
     CONSTRAINT fk_prof_dep FOREIGN KEY (chair_prof_ssn) REFERENCES clg.Professor(prof_ssn) ON DELETE CASCADE
  );

  --Students Table
  CREATE TABLE clg.Student
  (
     stu_ssn CHAR(9) NOT NULL,
     stu_name VARCHAR(300) NOT NULL,
     stu_age INT NULL,
     deg_program CHAR(3) NOT NULL CHECK(deg_program IN ('MS', 'PhD')),
     dept_no INT NULL,
     advisor_stu_ssn CHAR(9) NOT NULL

     CONSTRAINT pk_stu_ssn PRIMARY KEY (stu_ssn),
     CONSTRAINT fk_stu_dept FOREIGN KEY (dept_no) REFERENCES clg.Department(dept_no)
  );

  --Project Managers Table
  CREATE TABLE clg.Project_Managers
  (
```

Question 5 – SKIP

LAB 1

Question 1

SQL in File: **Lab1_Answers.sql**

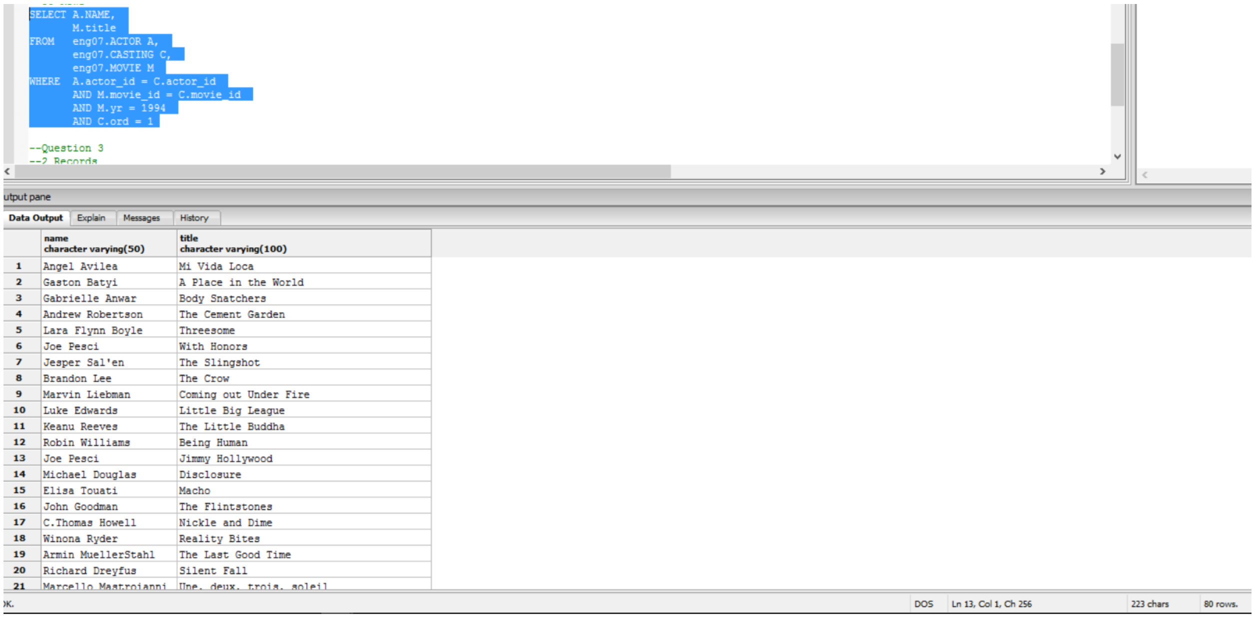


```
--Question 1
--9 Records
SELECT A.NAME
FROM   eng07.ACTOR A,
       eng07.CASTING C,
       eng07.MOVIE M
WHERE  A.actor_id = C.actor_id
       AND C.movie_id = M.movie_id
       AND M.title = 'The Great Gatsby'
```

	name character varying(50)
1	Alan Ladd
2	Macdonald Carey
3	Betty Field
4	Barry Sullivan
5	Howard daSilva
6	Robert Redford
7	Mia Farrow
8	Bruce Dern
9	Karen Black

Question 2

SQL in File: **Lab1_Answers.sql**



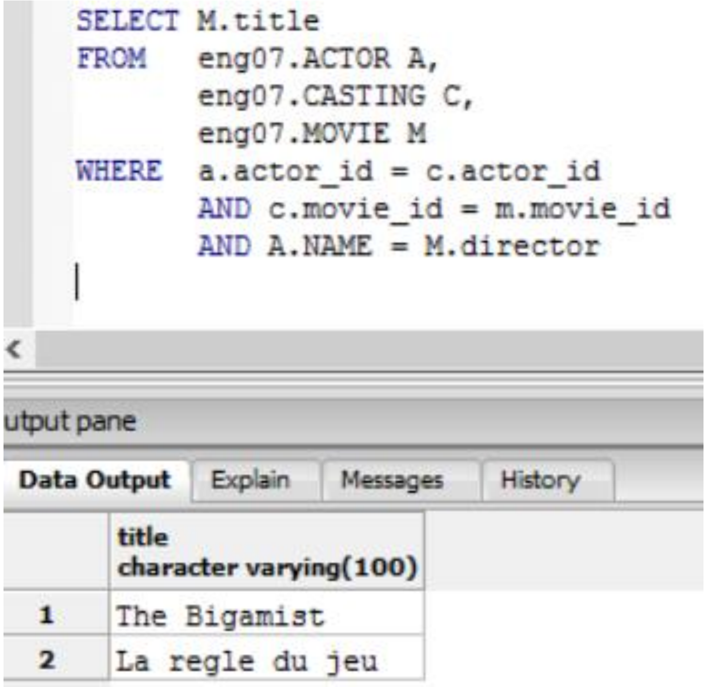
```
SELECT A.NAME,
       M.title
FROM   eng07.ACTOR A,
       eng07.CASTING C,
       eng07.MOVIE M
WHERE  A.actor_id = C.actor_id
       AND M.movie_id = C.movie_id
       AND M.yr = 1994
       AND C.ord = 1

--Question 3
--2 Records
```

	name character varying(50)	title character varying(100)
1	Angel Avilea	Mi Vida Loca
2	Gaston Batyi	A Place in the World
3	Gabrielle Anwar	Body Snatchers
4	Andrew Robertson	The Cement Garden
5	Lara Flynn Boyle	Threesome
6	Joe Pesci	With Honors
7	Jesper Sal'en	The Slingshot
8	Brandon Lee	The Crow
9	Marvin Liebman	Coming out Under Fire
10	Luke Edwards	Little Big League
11	Keanu Reeves	The Little Buddha
12	Robin Williams	Being Human
13	Joe Pesci	Jimmy Hollywood
14	Michael Douglas	Disclosure
15	Elisa Touati	Macho
16	John Goodman	The Flintstones
17	C.Thomas Howell	Nickle and Dime
18	Winona Ryder	Reality Bites
19	Armin MuellerStahl	The Last Good Time
20	Richard Dreyfus	Silent Fall
21	Marcello Mastroianni	Une, deux, trois, soleil

```
)K.                DOS   Ln 13, Col 1, Ch 256        223 chars    80 rows.
```

Question 3

SQL in File: **Lab1_Answers.sql**



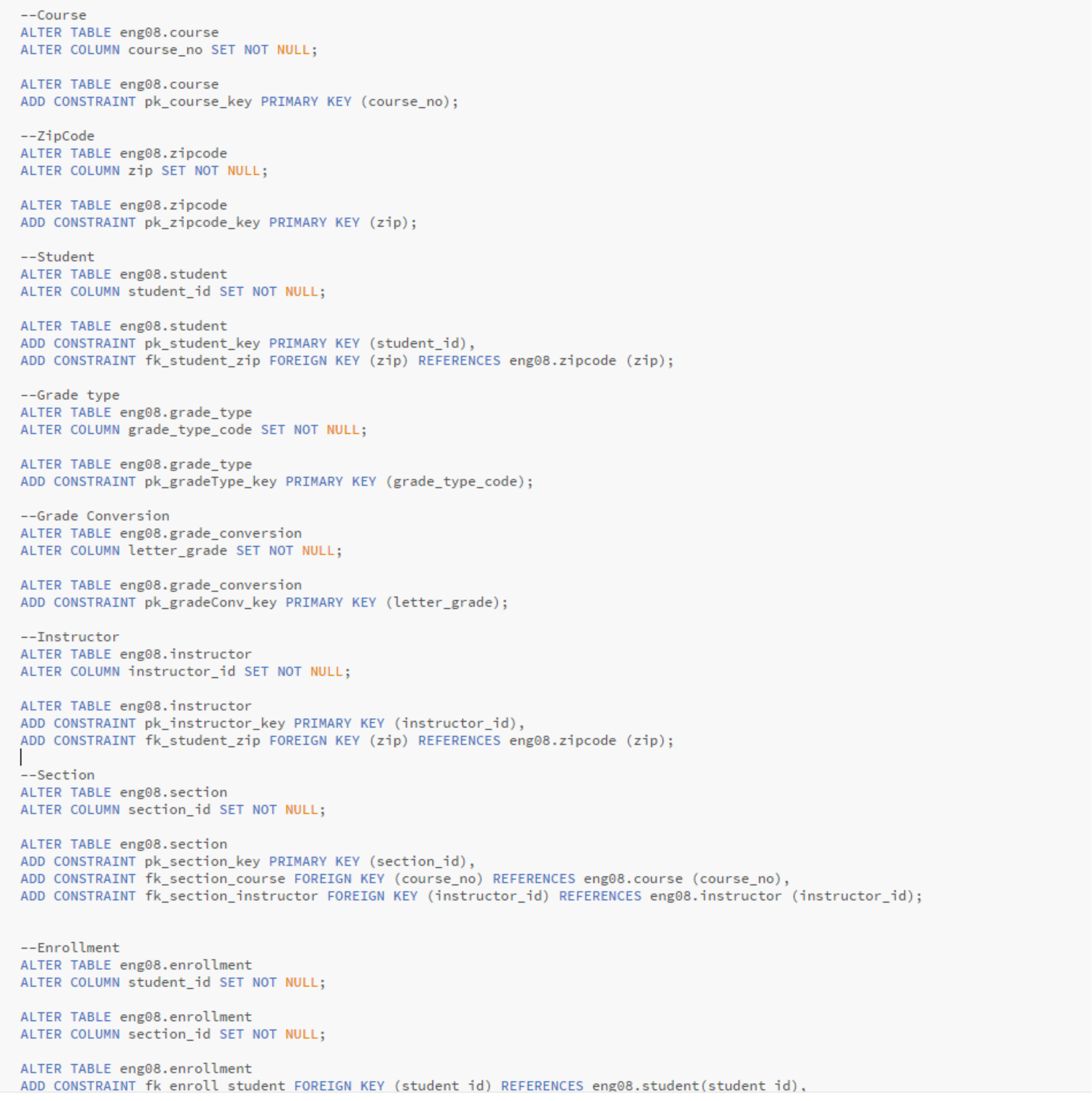
```
SELECT M.title
FROM   eng07.ACTOR A,
       eng07.CASTING C,
       eng07.MOVIE M
WHERE  a.actor_id = c.actor_id
       AND c.movie_id = m.movie_id
       AND A.NAME = M.director
```

	title character varying(100)
1	The Bigamist
2	La regle du jeu

LAB 2

Question 4

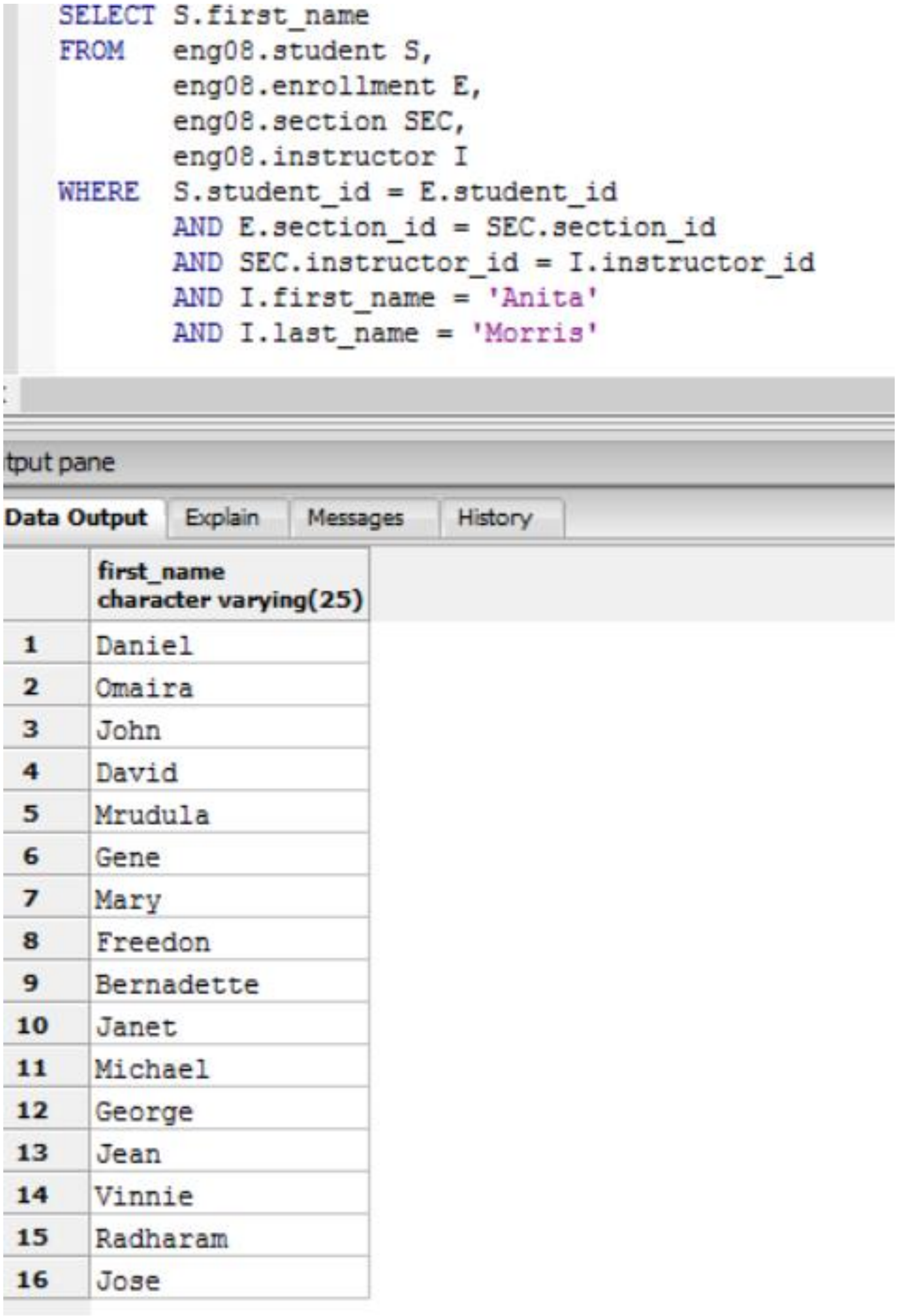
SQL in File: Lab2_Q4_eng08-relations-POSTGRE



```
--Course
ALTER TABLE eng08.course
ALTER COLUMN course_no SET NOT NULL;

ALTER TABLE eng08.course
ADD CONSTRAINT pk_course_key PRIMARY KEY (course_no);

--ZipCode
ALTER TABLE eng08.zipcode
ALTER COLUMN zip SET NOT NULL;

ALTER TABLE eng08.zipcode
ADD CONSTRAINT pk_zipcode_key PRIMARY KEY (zip);

--Student
ALTER TABLE eng08.student
ALTER COLUMN student_id SET NOT NULL;

ALTER TABLE eng08.student
ADD CONSTRAINT pk_student_key PRIMARY KEY (student_id),
ADD CONSTRAINT fk_student_zip FOREIGN KEY (zip) REFERENCES eng08.zipcode (zip);

--Grade type
ALTER TABLE eng08.grade_type
ALTER COLUMN grade_type_code SET NOT NULL;

ALTER TABLE eng08.grade_type
ADD CONSTRAINT pk_gradeType_key PRIMARY KEY (grade_type_code);

--Grade Conversion
ALTER TABLE eng08.grade_conversion
ALTER COLUMN letter_grade SET NOT NULL;

ALTER TABLE eng08.grade_conversion
ADD CONSTRAINT pk_gradeConv_key PRIMARY KEY (letter_grade);

--Instructor
ALTER TABLE eng08.instructor
ALTER COLUMN instructor_id SET NOT NULL;

ALTER TABLE eng08.instructor
ADD CONSTRAINT pk_instructor_key PRIMARY KEY (instructor_id),
ADD CONSTRAINT fk_student_zip FOREIGN KEY (zip) REFERENCES eng08.zipcode (zip);

--Section
ALTER TABLE eng08.section
ALTER COLUMN section_id SET NOT NULL;

ALTER TABLE eng08.section
ADD CONSTRAINT pk_section_key PRIMARY KEY (section_id),
ADD CONSTRAINT fk_section_course FOREIGN KEY (course_no) REFERENCES eng08.course (course_no),
ADD CONSTRAINT fk_section_instructor FOREIGN KEY (instructor_id) REFERENCES eng08.instructor (instructor_id);


--Enrollment
ALTER TABLE eng08.enrollment
ALTER COLUMN student_id SET NOT NULL;

ALTER TABLE eng08.enrollment
ALTER COLUMN section_id SET NOT NULL;

ALTER TABLE eng08.enrollment
ADD CONSTRAINT fk_enroll_student FOREIGN KEY (student_id) REFERENCES eng08.student(student_id),
```

Question 5

SQL in File: **Lab2_Answers.sql**

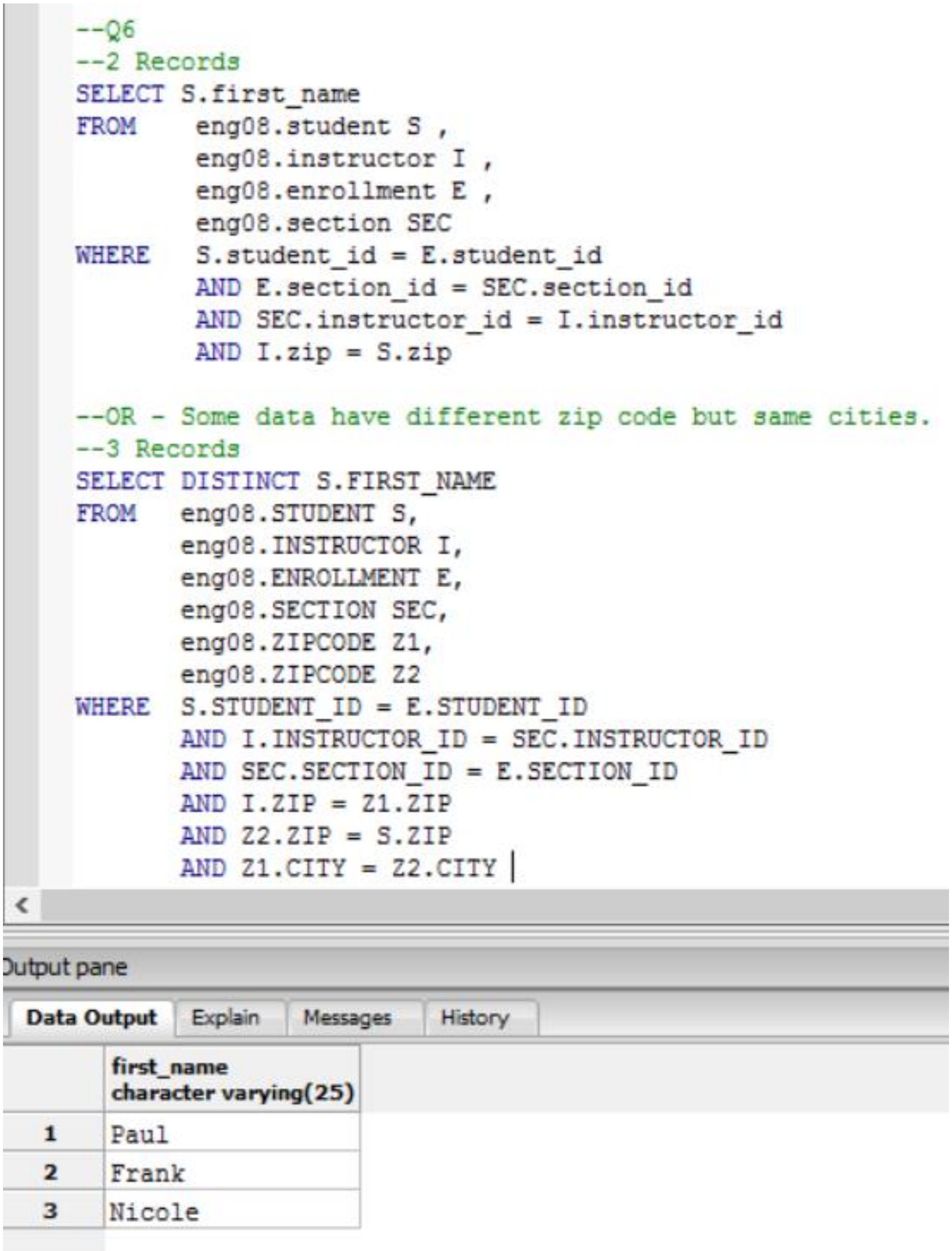


```
SELECT S.first_name
FROM   eng08.student S,
       eng08.enrollment E,
       eng08.section SEC,
       eng08.instructor I
WHERE  S.student_id = E.student_id
       AND E.section_id = SEC.section_id
       AND SEC.instructor_id = I.instructor_id
       AND I.first_name = 'Anita'
       AND I.last_name = 'Morris'
```

	first_name character varying(25)
1	Daniel
2	Omaira
3	John
4	David
5	Mrudula
6	Gene
7	Mary
8	Freedon
9	Bernadette
10	Janet
11	Michael
12	George
13	Jean
14	Vinnie
15	Radharam
16	Jose

Question 6

SQL in File: **Lab2_Answers.sql**

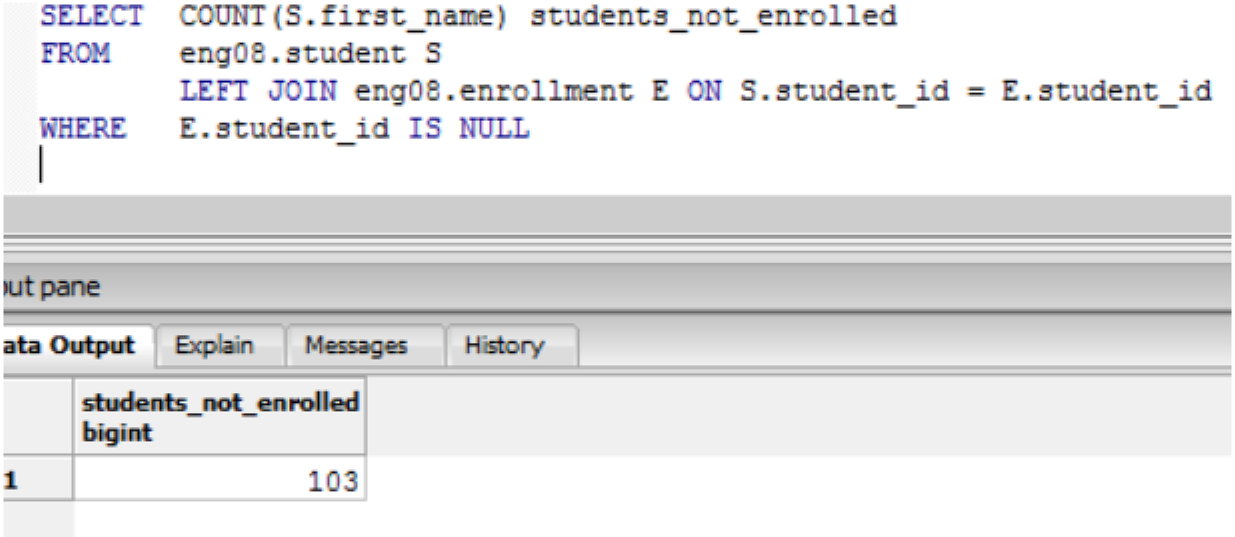


```
--Q6
--2 Records
SELECT S.first_name
FROM    eng08.student S ,
        eng08.instructor I ,
        eng08.enrollment E ,
        eng08.section SEC
WHERE   S.student_id = E.student_id
        AND E.section_id = SEC.section_id
        AND SEC.instructor_id = I.instructor_id
        AND I.zip = S.zip

--OR - Some data have different zip code but same cities.
--3 Records
SELECT DISTINCT S.FIRST_NAME
FROM   eng08.STUDENT S,
       eng08.INSTRUCTOR I,
       eng08.ENROLLMENT E,
       eng08.SECTION SEC,
       eng08.ZIPCODE Z1,
       eng08.ZIPCODE Z2
WHERE  S.STUDENT_ID = E.STUDENT_ID
       AND I.INSTRUCTOR_ID = SEC.INSTRUCTOR_ID
       AND SEC.SECTION_ID = E.SECTION_ID
       AND I.ZIP = Z1.ZIP
       AND Z2.ZIP = S.ZIP
       AND Z1.CITY = Z2.CITY
```

```
     first_name
     character varying(25)
1    Paul
2    Frank
3    Nicole
```

Question 7

SQL in File: **Lab2_Answers.sql**

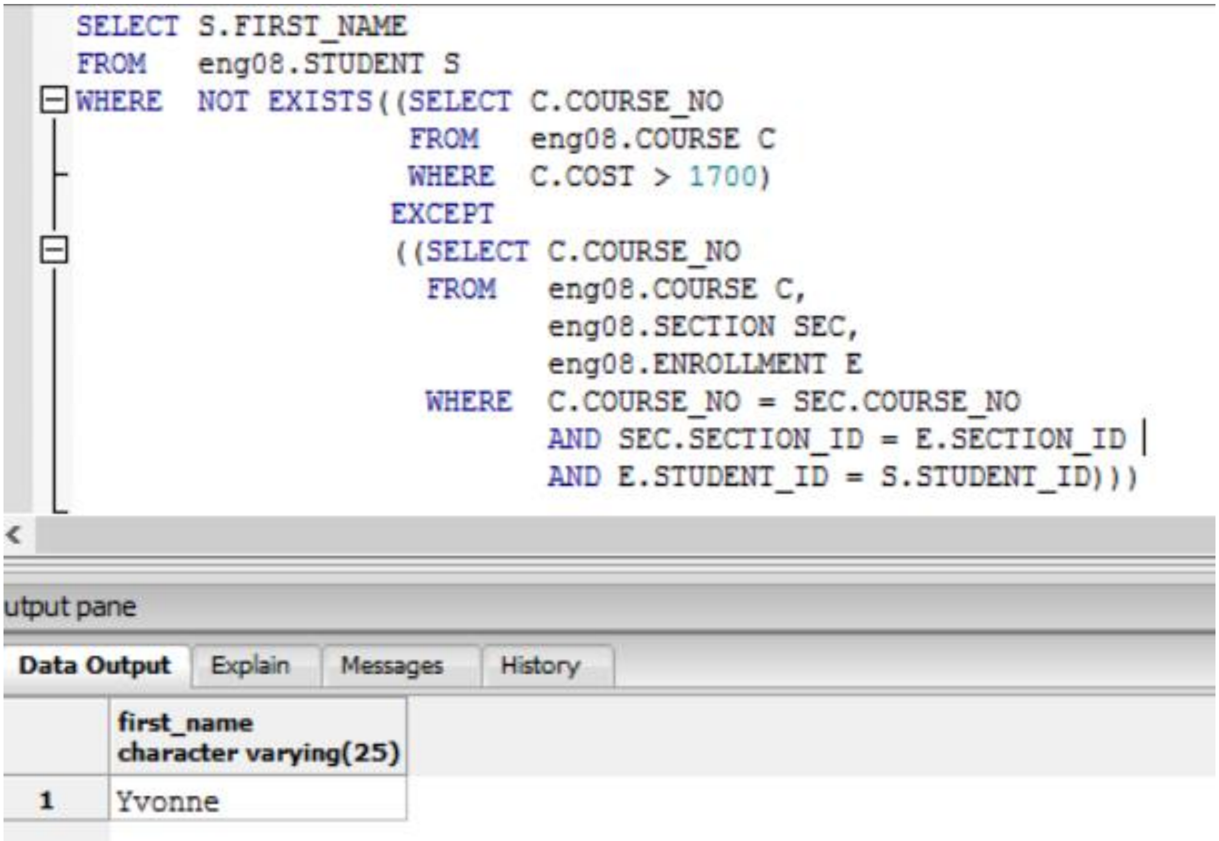


```sql
SELECT  COUNT(S.first_name) students_not_enrolled
FROM    eng08.student S
        LEFT JOIN eng08.enrollment E ON S.student_id = E.student_id
WHERE   E.student_id IS NULL
```

	students_not_enrolled bigint
1	103

Question 8

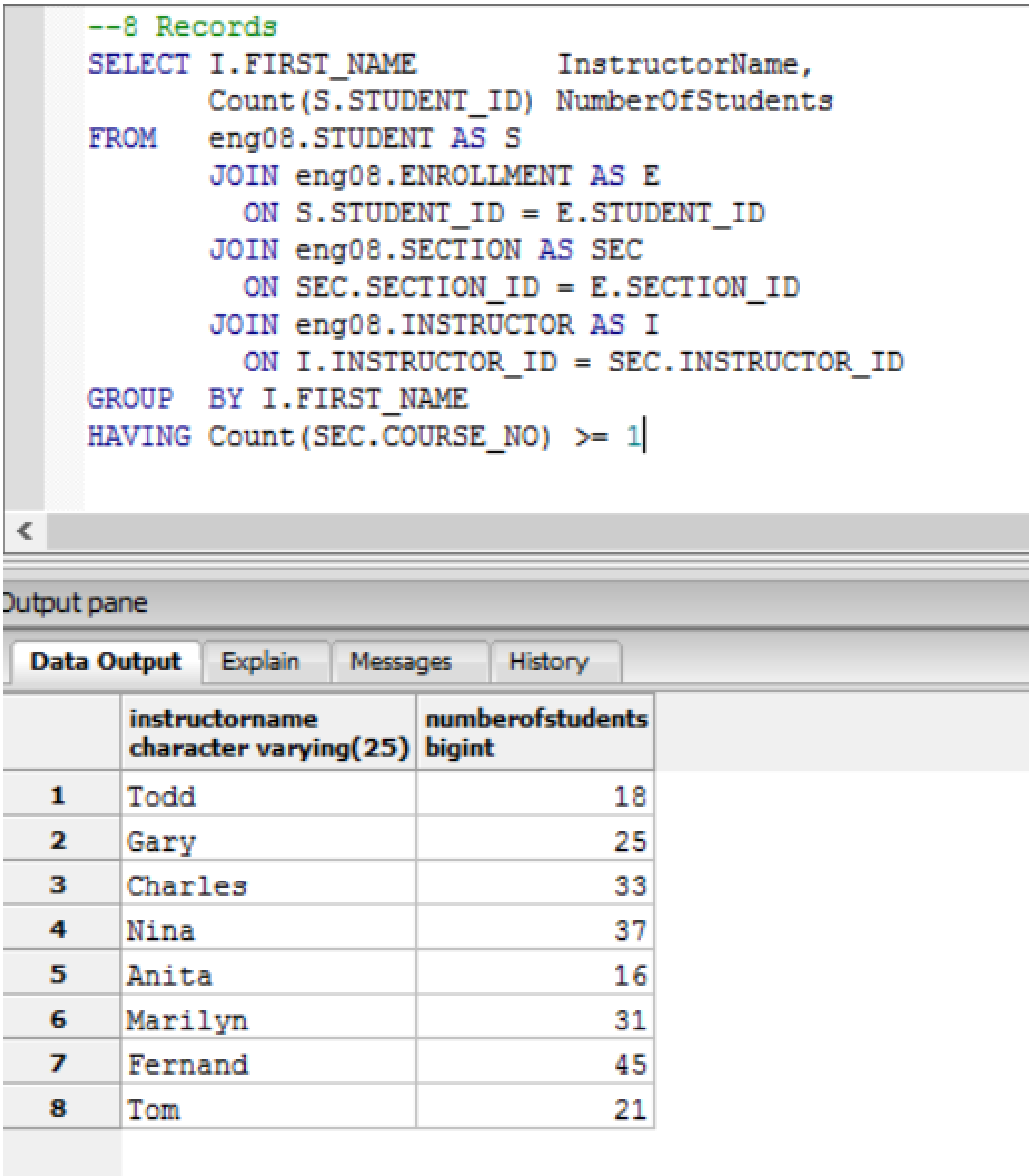
SQL in File: **Lab2_Answers.sql**



```sql
SELECT S.FIRST_NAME
FROM   eng08.STUDENT S
WHERE  NOT EXISTS((SELECT C.COURSE_NO
                   FROM   eng08.COURSE C
                   WHERE  C.COST > 1700)
                  EXCEPT
                  ((SELECT C.COURSE_NO
                    FROM   eng08.COURSE C,
                           eng08.SECTION SEC,
                           eng08.ENROLLMENT E
                    WHERE  C.COURSE_NO = SEC.COURSE_NO
                           AND SEC.SECTION_ID = E.SECTION_ID
                           AND E.STUDENT_ID = S.STUDENT_ID)))
```

	first_name character varying(25)
1	Yvonne

Question 9



```
--8 Records
SELECT I.FIRST_NAME         InstructorName,
       Count(S.STUDENT_ID) NumberOfStudents
FROM   eng08.STUDENT AS S
       JOIN eng08.ENROLLMENT AS E
         ON S.STUDENT_ID = E.STUDENT_ID
       JOIN eng08.SECTION AS SEC
         ON SEC.SECTION_ID = E.SECTION_ID
       JOIN eng08.INSTRUCTOR AS I
         ON I.INSTRUCTOR_ID = SEC.INSTRUCTOR_ID
GROUP  BY I.FIRST_NAME
HAVING Count(SEC.COURSE_NO) >= 1
```

	instructorname character varying(25)	numberofstudents bigint
1	Todd	18
2	Gary	25
3	Charles	33
4	Nina	37
5	Anita	16
6	Marilyn	31
7	Fernand	45
8	Tom	21