Lab Assignment 3

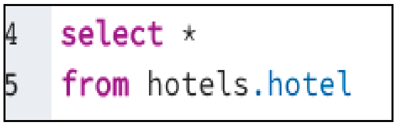
Table of Contents

Exercise 6.7

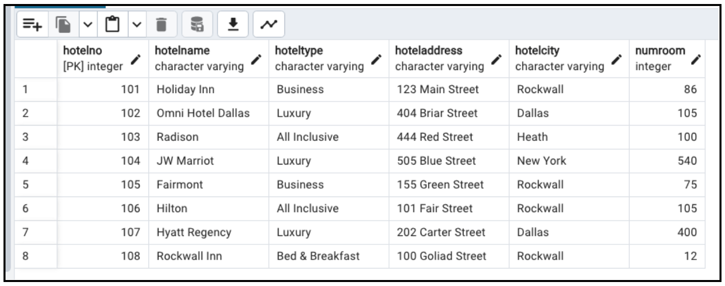
Consider the following Hotel, Room, Booking and Guest schemas in a DBMS. The *hotelNo* is the primary key for the Hotel table and *roomNo* is the primary key for the Room relation. Booking stores the details of room reservations and *bookingNo* is the primary key. Guest stores the guests details and *guestNo* is the primary key.

Hotel (*hotelNo*, hotelName, hotelType, hotelAddress, hotelCity, numRoom)
Room (*roomNo*, hotelNo, roomPrice)
Booking (*bookingNo*, hotelNo, guestNo, checkIn, checkout, totelGuest, roomNo)
Guest (*guestNo*, firstName, lastName, guestAddress)

1. **Write the SQL to list full details of all the hotels:**

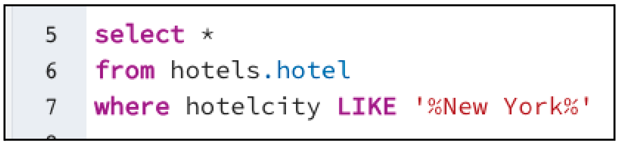


```
select *
from hotels.hotel
```

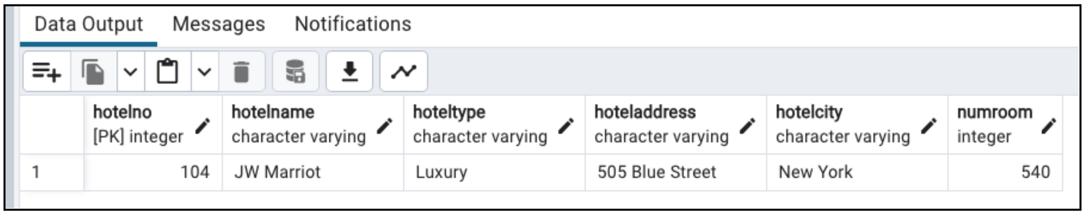


	hotelno [PK] integer	hotelname character varying	hoteltype character varying	hoteladdress character varying	hotelcity character varying	numroom integer
1	101	Holiday Inn	Business	123 Main Street	Rockwall	86
2	102	Omni Hotel Dallas	Luxury	404 Briar Street	Dallas	105
3	103	Radison	All Inclusive	444 Red Street	Heath	100
4	104	JW Marriot	Luxury	505 Blue Street	New York	540
5	105	Fairmont	Business	155 Green Street	Rockwall	75
6	106	Hilton	All Inclusive	101 Fair Street	Rockwall	105
7	107	Hyatt Regency	Luxury	202 Carter Street	Dallas	400
8	108	Rockwall Inn	Bed & Breakfast	100 Goliad Street	Rockwall	12

2. **Write the SQL to list full details of all the hotels in New York:**

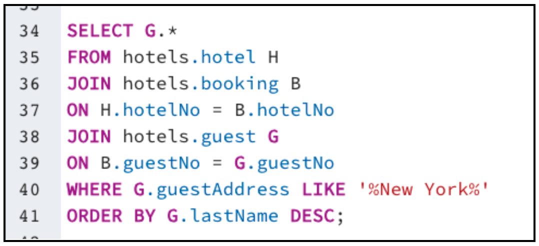


```
select *
from hotels.hotel
where hotelcity LIKE '%New York%'
```

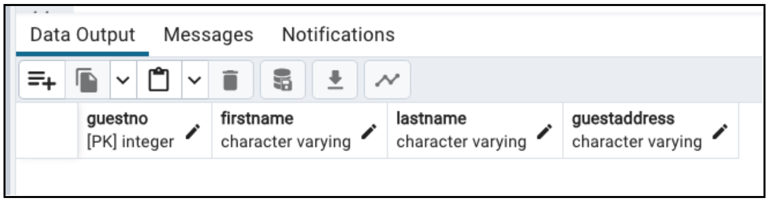


	hotelno [PK] integer	hotelname character varying	hoteltype character varying	hoteladdress character varying	hotelcity character varying	numroom integer
1	104	JW Marriot	Luxury	505 Blue Street	New York	540

3. **Write the SQL to list the guests in New York in descending order by last name.**



```
SELECT G.*
FROM hotels.hotel H
JOIN hotels.booking B
ON H.hotelNo = B.hotelNo
JOIN hotels.guest G
ON B.guestNo = G.guestNo
WHERE G.guestAddress LIKE '%New York%'
ORDER BY G.lastName DESC;
```



guestno [PK] integer	firstname character varying	lastname character varying	guestaddress character varying

Exercise 6.8

Write appropriate SQL DDL statements for declaring the LIBRARY relational database schema of Figure 6.6. Specify the keys and referential triggered actions.

Write the schema create statement along with the relation create statements. You may insert data, but this is optional. Please submit your SQL DDL (schema & tables).

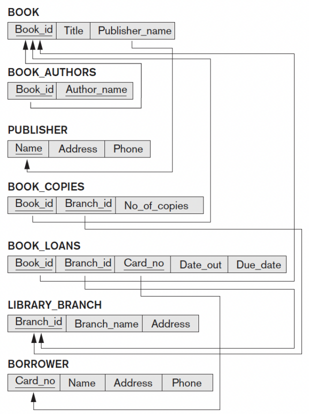
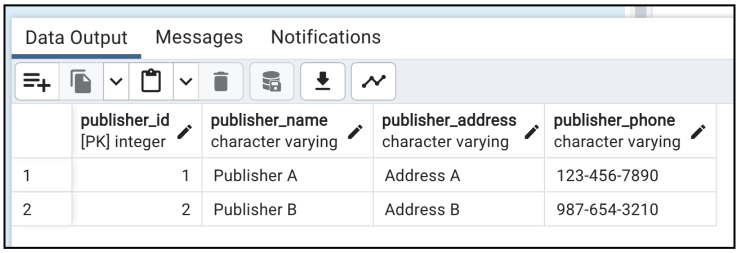


Figure 6.6
A relational database schema for a LIBRARY database.

```
DROP TABLE IF EXISTS libraries.book;
DROP TABLE IF EXISTS libraries.book_authors;
DROP TABLE IF EXISTS libraries.book_copies;
DROP TABLE IF EXISTS libraries.book_loans;
DROP TABLE IF EXISTS libraries.borrower;
DROP TABLE IF EXISTS libraries.library_branch;
DROP TABLE IF EXISTS libraries.publisher;

DROP SCHEMA IF EXISTS libraries;

CREATE SCHEMA IF NOT EXISTS libraries;

--Create the book table

CREATE TABLE IF NOT EXISTS libraries.book (
        book_id SERIAL,
        title VARCHAR NOT NULL,
        publisher_id INT,
        PRIMARY KEY(book_id)
        );

--Create the publisher table
CREATE TABLE IF NOT EXISTS libraries.publisher (
        publisher_id SERIAL,
        publisher_name VARCHAR NOT NULL,
        publisher_address VARCHAR NOT NULL,
        publisher_phone VARCHAR NOT NULL,
        PRIMARY KEY(publisher_id)
);

--Create the book_authors table
CREATE TABLE IF NOT EXISTS libraries.book_authors (
        book_id INT,
        author_name VARCHAR NOT NULL,
        FOREIGN KEY(book_id)
        REFERENCES libraries.book(book_id)
);

--Create the library_branch table
CREATE TABLE IF NOT EXISTS libraries.library_branch (
            branch_id SERIAL,
            branch_name VARCHAR NOT NULL,
            branch_address VARCHAR NOT NULL,
            PRIMARY KEY (branch_id)
);

--Create the book_copies table
CREATE TABLE IF NOT EXISTS libraries.book_copies (
            book_id INT,
            branch_id INT,
            no_of_copies INT NOT NULL,
            PRIMARY KEY (book_id, branch_id),
   FOREIGN KEY (book_id) REFERENCES libraries.book(book_id),
   FOREIGN KEY (branch_id) REFERENCES libraries.library_branch(branch_id)
);

--Create the borrower table
CREATE TABLE IF NOT EXISTS libraries.borrower (
            card_no SERIAL,
            borrower_name VARCHAR NOT NULL,
            borrower_address VARCHAR NOT NULL,
            borrower_phone VARCHAR NOT NULL,
            PRIMARY KEY (card_no)
);

--Create the book_loans table
CREATE TABLE IF NOT EXISTS libraries.book_loans (
            loan_id SERIAL,
            book_id INT,
            branch_id INT,
            card_no INT,
            date_out DATE NOT NULL,
            due_date DATE NOT NULL,
            PRIMARY KEY (loan_id),
            FOREIGN KEY (book_id, branch_id) REFERENCES libraries.book_copies(book_id, branch_id),
   FOREIGN KEY (card_no) REFERENCES libraries.borrower(card_no)
);

-- Optional: Insert data into the publisher table
INSERT INTO libraries.publisher (publisher_name, publisher_address, publisher_phone)
VALUES
   ('Publisher A', 'Address A', '123-456-7890'),
   ('Publisher B', 'Address B', '987-654-3210');

-- Optional: Insert data into the library_branch table
INSERT INTO libraries.library_branch (branch_name, branch_address)
```

```
VALUES
  ('Branch 1', 'Branch Address 1'),
  ('Branch 2', 'Branch Address 2');
```

```
SELECT * from libraries.publisher
```



	publisher_id [PK] integer	publisher_name character varying	publisher_address character varying	publisher_phone character varying
1	1	Publisher A	Address A	123-456-7890
2	2	Publisher B	Address B	987-654-3210

In the provided SQL code, I have defined the schema and tables for a library database. Here are the keys and referential triggered actions for the tables:

1. libraries.book table:
   a. Primary Key: book_id
   b. No referential actions specified.
2. libraries.publisher table:
   a. Primary Key: publisher_id
   b. No referential actions specified.
3. libraries.book_authors table:
   a. No primary key specified (this may not be ideal; consider adding a primary key).
   b. Foreign Key: book_id references libraries.book(book_id).
4. libraries.library_branch table:
   a. Primary Key: branch_id
   b. No referential actions specified.
5. libraries.book_copies table:
   a. Primary Key: (book_id, branch_id)
   b. Foreign Key: book_id references libraries.book(book_id)
   c. Foreign Key: branch_id references libraries.library_branch(branch_id)
6. libraries.borrower table:
   a. Primary Key: card_no
   b. No referential actions specified.
7. libraries.book_loans table:
   a. Primary Key: loan_id
   b. Foreign Key: (book_id, branch_id) references libraries.book_copies(book_id, branch_id)
   c. Foreign Key: card_no references libraries.borrower(card_no)

In this schema:

1. The primary keys are explicitly defined for each table.
2. Foreign keys are used to establish relationships between tables to enforce referential integrity.
3. There are no specific referential triggered actions (such as CASCADE or SET NULL) defined in the SQL code. This means that when a referenced record is updated or deleted, the default behavior of your DBMS will be used (usually restricting the action unless specified otherwise).

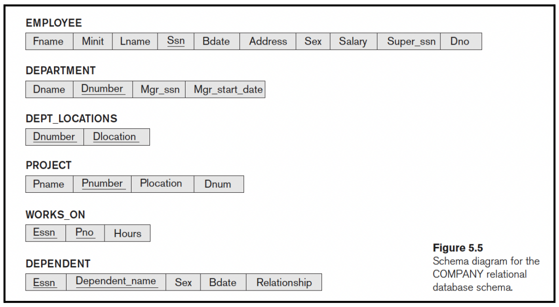
Please note that the libraries.book_authors table doesn't have an explicit primary key. It's a good practice to have a primary key in every table to ensure data integrity and efficient querying. We could consider adding an additional column like author_id as a primary key in that table, or use a composite primary key if applicable.

Exercise 6.10

Specify the following queries in SQL on the COMPANY relational database schema shown in Figure 5.5. Show the result of each query if it is applied to the COMPANY database in Figure 5.6. You will need to create the INSERT statements to match the data in figure 5.5 (page 191 & 192).

1. Retrieve the names of all employees in department 5 who earn more than 3000 and work on the ProductZ project.
2. List the names of all employees who are from Houston, Texas and work under manager 333445555.
3. Find the names of all employees who are working in the project Computerization.

Please submit your DDL (schema and tables), queries, and query results.



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Figure 5.5
Schema diagram for the COMPANY relational database schema.

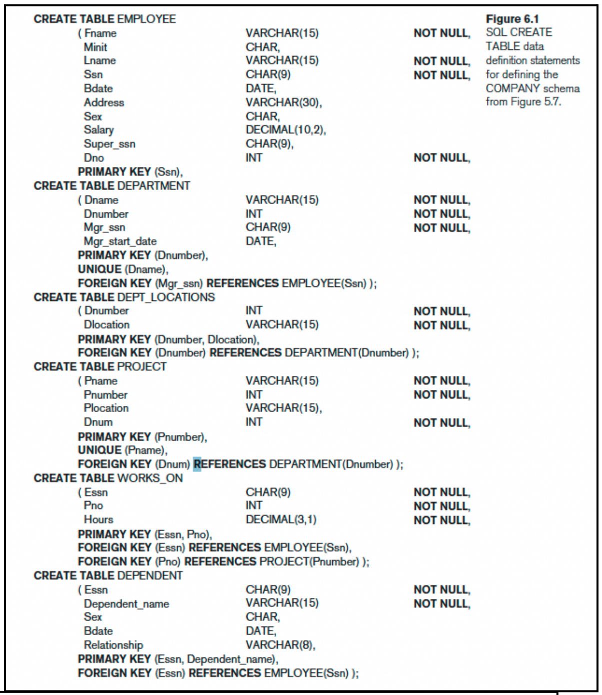


Figure 6.1
SQL CREATE TABLE data definition statements for defining the COMPANY schema from Figure 5.7.

```
CREATE TABLE EMPLOYEE
        ( Fname                 VARCHAR(15)             NOT NULL,
          Minit                 CHAR,
          Lname                 VARCHAR(15)             NOT NULL,
          Ssn                   CHAR(9)                 NOT NULL,
          Bdate                 DATE,
          Address               VARCHAR(30),
          Sex                   CHAR,
          Salary                DECIMAL(10,2),
          Super_ssn             CHAR(9),
          Dno                   INT                     NOT NULL,
        PRIMARY KEY (Ssn),
CREATE TABLE DEPARTMENT
        ( Dname                 VARCHAR(15)             NOT NULL,
          Dnumber               INT                     NOT NULL,
          Mgr_ssn               CHAR(9)                 NOT NULL,
          Mgr_start_date        DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
CREATE TABLE DEPT_LOCATIONS
        ( Dnumber               INT                     NOT NULL,
          Dlocation             VARCHAR(15)             NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE PROJECT
        ( Pname                 VARCHAR(15)             NOT NULL,
          Pnumber               INT                     NOT NULL,
          Plocation             VARCHAR(15),
          Dnum                  INT                     NOT NULL,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE WORKS_ON
        ( Essn                  CHAR(9)                 NOT NULL,
          Pno                   INT                     NOT NULL,
          Hours                 DECIMAL(3,1)            NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn                  CHAR(9)                 NOT NULL,
          Dependent_name        VARCHAR(15)             NOT NULL,
          Sex                   CHAR,
          Bdate                 DATE,
          Relationship          VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

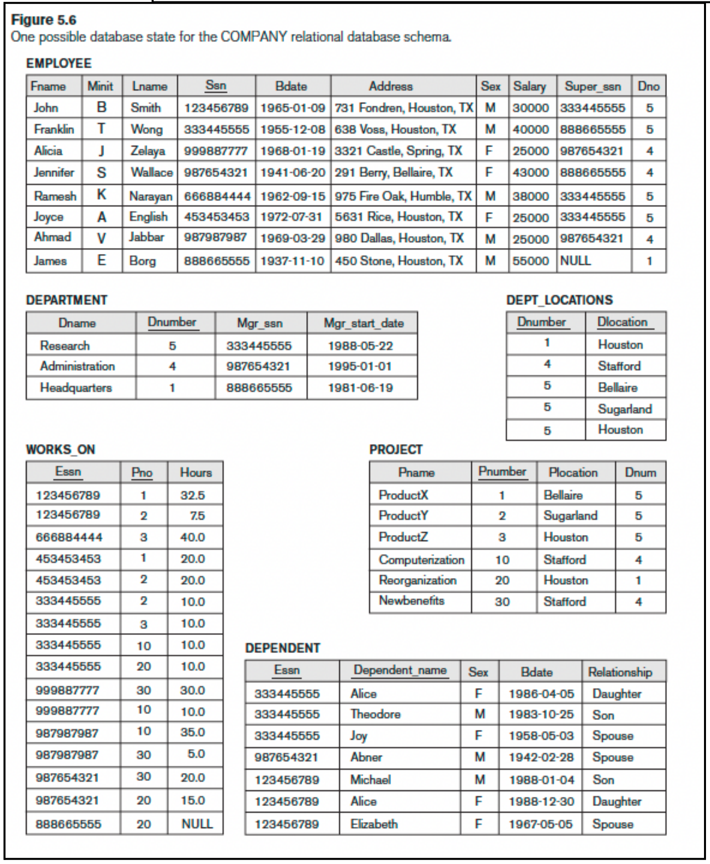


Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

```
DROP TABLE IF EXISTS company.employee;
DROP TABLE IF EXISTS company.department;
DROP TABLE IF EXISTS company.dept_locations;
DROP TABLE IF EXISTS company.project;
DROP TABLE IF EXISTS company.works_on;
DROP TABLE IF EXISTS company.dependent;
DROP TABLE IF EXISTS libraries.publisher;

DROP SCHEMA IF EXISTS company;
CREATE SCHEMA IF NOT EXISTS company;

-- Create EMPLOYEE table in the company schema
CREATE TABLE company.EMPLOYEE (
  Fname VARCHAR(15),
  Minit CHAR,
  Lname VARCHAR(15),
  Ssn CHAR(9) PRIMARY KEY,
  Bdate DATE,
  Address VARCHAR(30),
  Sex CHAR,
  Salary DECIMAL(10, 2),
  Super_ssn CHAR(9),
  Dno INT NOT NULL
);

-- Create DEPARTMENT table in the company schema
CREATE TABLE company.DEPARTMENT (
  Dname VARCHAR(15),
  Dnumber INT NOT NULL PRIMARY KEY,
```

```sql
   Mgr_ssn CHAR(9) NOT NULL,
   Mgr_start_date DATE,
   FOREIGN KEY (Mgr_ssn) REFERENCES company.EMPLOYEE(Ssn)
);

-- Create DEPT_LOCATIONS table in the company schema
CREATE TABLE company.DEPT_LOCATIONS (
   Dnumber INT NOT NULL,
   Dlocation VARCHAR(15) NOT NULL,
   PRIMARY KEY (Dnumber, Dlocation),
   FOREIGN KEY (Dnumber) REFERENCES company.DEPARTMENT(Dnumber)
);

-- Create PROJECT table in the company schema
CREATE TABLE company.PROJECT (
   Pname VARCHAR(15),
   Pnumber INT NOT NULL PRIMARY KEY,
   Plocation VARCHAR(15),
   Dnum INT NOT NULL,
   FOREIGN KEY (Dnum) REFERENCES company.DEPARTMENT(Dnumber)
);

-- Create WORKS_ON table in the company schema
CREATE TABLE company.WORKS_ON (
   Essn CHAR(9) NOT NULL,
   Pno INT NOT NULL,
   Hours DECIMAL(3, 1) NOT NULL,
   PRIMARY KEY (Essn, Pno),
   FOREIGN KEY (Essn) REFERENCES company.EMPLOYEE(Ssn),
   FOREIGN KEY (Pno) REFERENCES company.PROJECT(Pnumber)
);

-- Create DEPENDENT table in the company schema
CREATE TABLE company.DEPENDENT (
   Essn CHAR(9) NOT NULL,
   Dependent_name VARCHAR(15) NOT NULL,
   Sex CHAR,
   Bdate DATE,
   Relationship VARCHAR(8),
   PRIMARY KEY (Essn, Dependent_name),
   FOREIGN KEY (Essn) REFERENCES company.EMPLOYEE(Ssn)
);

INSERT INTO company.employee (Fname, Minit, Lname, Ssn, Bdate, Address, Sex, Salary, Super_ssn, Dno)
VALUES
   ('John', 'B', 'Smith', '123456789', '1965-01-09', '731 Fondren, Houston, TX', 'M', 30000, '333445555', 5),
   ('Franklin', 'T', 'Wong', '333445555', '1955-12-08', '638 Voss, Houston, TX', 'M', 40000, '888665555', 5),
   ('Alicia', 'J', 'Zelaya', '999887777', '1968-01-19', '3321 Castle, Spring, TX', 'F', 25000, '987654321', 4),
   ('Jennifer', 'S', 'Wallace', '987654321', '1941-06-20', '291 Berry, Bellaire,TX', 'F', 43000, '888665555', 4),
   ('Ramesh', 'K', 'Narayan', '666884444', '1962-09-15', '975 Fire Oak, Humble,TX', 'M', 38000, '333445555', 5),
   ('Joyce', 'A', 'English', '453453453', '1972-07-31', '5631 Rice, Houston, TX', 'F', 25000, '333445555', 5),
   ('Ahmad', 'V', 'Jabbar', '987987987', '1969-03-29', '980 Dallas, Houston, TX', 'M', 25000, '987654321', 4),
   ('James', 'E', 'Borg', '888665555', '1937-11-10', '450 Stone, Houston, TX', 'M', 55000, NULL, 1);

INSERT INTO company.department (Dname, Dnumber, Mgr_ssn, Mgr_start_date)
VALUES
   ('Research', 5, '333445555', '1988-05-22'),
   ('Administration', 4, '987654321', '1995-01-01'),
   ('Headquarters', 1, '888665555', '1981-06-19');

INSERT INTO company.dept_locations (Dnumber, Dlocation)
VALUES
   (1, 'Houston'),
   (4, 'Stafford'),
   (5, 'Bellaire'),
   (5, 'Sugarland'),
   (1, 'Houston');

INSERT INTO company.project (Pname, Pnumber, Plocation, Dnum)
VALUES
   ('ProductX', 1, 'Bellaire', 5),
   ('ProductY', 2, 'Sugarland', 5),
   ('ProductZ', 3, 'Houston', 5),
   ('Computerization', 10, 'Stafford', 4),
   ('Reorganization', 20, 'Houston', 1),
   ('Newbenefits', 30, 'Stafford', 4);

INSERT INTO company.works_on (Essn, Pno, Hours)
VALUES
   ('123456789', 1, 32.5),
   ('123456789', 2, 7.5),
   ('666884444', 3, 40.0),
   ('453453453', 1, 20.0),
   ('453453453', 2, 20.0),
   ('333445555', 2, 10.0),
   ('333445555', 3, 10.0),
   ('333445555', 10, 10.0),
   ('333445555', 20, 10.0),
   ('999887777', 30, 30.0),
   ('999887777', 10, 10.0),
   ('987987987', 10, 35.0),
   ('987987987', 30, 5.0),
   ('987654321', 30, 20.0),
   ('987654321', 20, 15.0),
   ('888665555', 20, 0);
INSERT INTO company.dependent (Essn, Dependent_name, Sex, Bdate, Relationship)
VALUES
   ('333445555', 'Alice', 'F', '1986-04-05', 'DAUGHTER'),
   ('333445555', 'Theodore', 'M', '1983-10-25', 'SON'),
   ('333445555', 'Joy', 'F', '1958-05-03', 'SPOUSE'),
   ('987654321', 'Abner', 'M', '1942-02-28', 'SPOUSE'),
   ('123456789', 'Michael', 'M', '1988-01-04', 'SON'),
   ('123456789', 'Alice', 'F', '1988-12-30', 'DAUGHTER'),
   ('123456789', 'Elizabeth', 'F', '1967-05-05', 'SPOUSE');
```

--a) Retrieve the names of all employees in department 5 who earn more than 3000 and work on the ProductZ project:

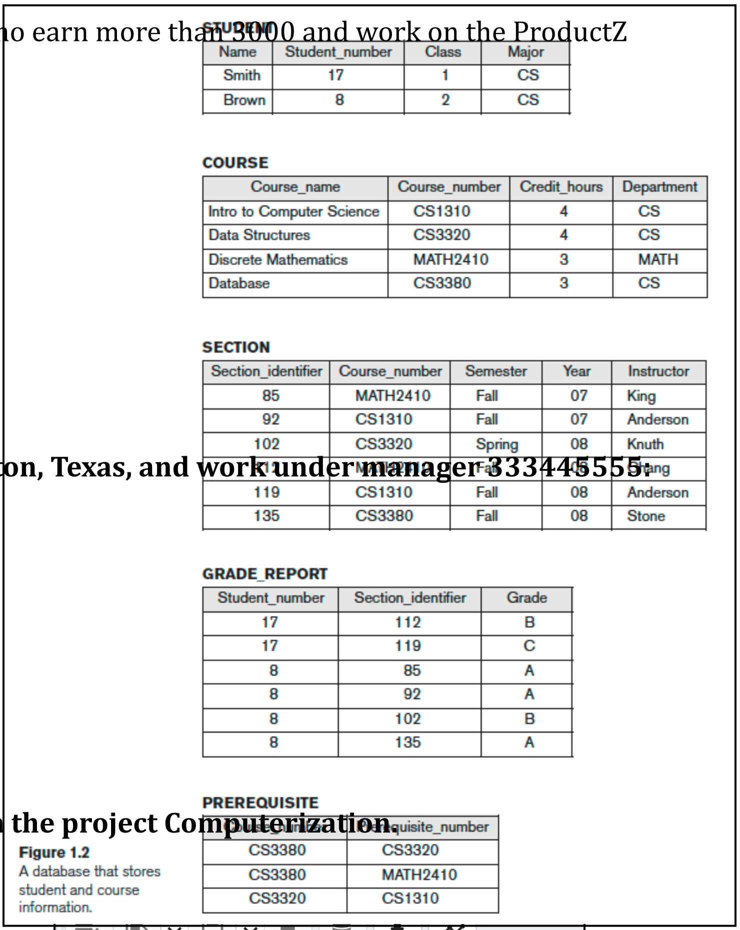
```
SELECT E.Fname, E.Lname
FROM company.employee E
JOIN company.department D ON E.Dno = D.Dnumber
JOIN company.works_on W ON E.Ssn = W.Essn
JOIN company.project P ON W.Pno = P.Pnumber
WHERE D.Dnumber = 5
  AND E.Salary > 3000
  AND P.Pname = 'ProductZ';
```

--b) List the names of all employees who are from Houston, Texas, and work under manager 333445555:

```
SELECT E.Fname, E.Lname
FROM company.employee E
JOIN company.department D ON E.Dno = D.Dnumber
WHERE D.Mgr_ssn = '333445555'
  AND E.Address LIKE '%Houston%'
  AND E.Address LIKE '%TX%';
```

--c) Find the names of all employees who are working in the project Computerization.

```
SELECT E.Fname, E.Lname
FROM company.employee E
JOIN company.works_on W ON E.Ssn = W.Essn
JOIN company.project P ON W.Pno = P.Pnumber
WHERE P.Pname = 'Computerization';
```



STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

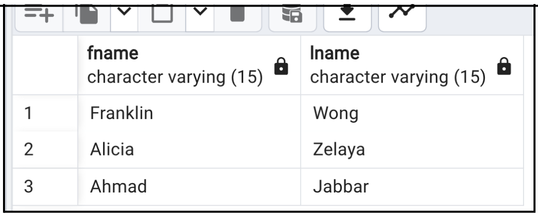
GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.



	fname character varying (15)	lname character varying (15)
1	Franklin	Wong
2	Alicia	Zelaya
3	Ahmad	Jabbar

Exercise 6.12

Specify the following queries in SQL on the database schema of Figure 1.2. (page 38)

1. Retrieve the course names of all the courses that come under the department of ‘cs’ (computer science).
2. Retrieve the names of all courses along with the name of the instructor taught during the fall of 2008.
3. For each section taught by Professor Anderson, retrieve the course number, semester, year, and number of students who took the section.
4. Retrieve the name and transcript of each junior student (Class = 1) majoring in mathematics (MATH). A transcript includes course name, course number, credit hours, semester, year, and grade for each course completed by the student.

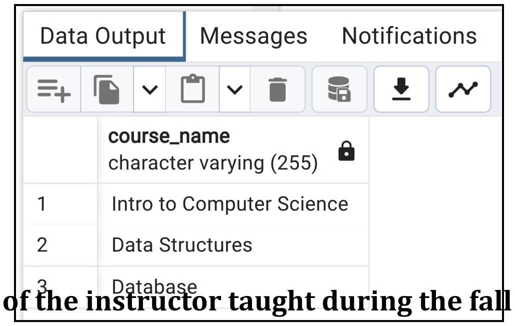
You MUST WRITE THE DDL and INSERT statements to create this schema and tables. Please submit your queries AND results. The DDL is needed for the final question.

```
DROP TABLE IF EXISTS school.student;
DROP TABLE IF EXISTS school.course;
DROP TABLE IF EXISTS school.section;
DROP TABLE IF EXISTS school.grade_report;
DROP TABLE IF EXISTS school.prerequisite;
DROP SCHEMA IF EXISTS school;

CREATE SCHEMA IF NOT EXISTS school;

CREATE TABLE school.student (
   Student_number INT PRIMARY KEY,
   Name VARCHAR(255),
   Class INT,
   Major VARCHAR(255)
);

CREATE TABLE school.course (
   Course_number VARCHAR(255) PRIMARY KEY,
   Course_name VARCHAR(255),
   Credit_hours INT,
   Department VARCHAR(255)
);

CREATE TABLE school.section (
   Section_identifier INT PRIMARY KEY,
   Course_number VARCHAR(255),
   Semester VARCHAR(255),
   Year INT,
   Instructor VARCHAR(255),
   FOREIGN KEY (Course_number) REFERENCES school.course(Course_number)
);

CREATE TABLE school.prerequisite (
   Course_number VARCHAR(255),
   Prerequisite_number VARCHAR(255),
   FOREIGN KEY (Course_number) REFERENCES school.course(Course_number),
   FOREIGN KEY (Prerequisite_number) REFERENCES school.course(Course_number)
);
```

```
-- Insert data into the Student table
INSERT INTO school.student (Name, Student_number, Class, Major)
VALUES
   ('Smith', 17, 1, 'CS'),
   ('Brown', 8, 2, 'CS');

-- Insert data into the Course table
INSERT INTO school.course (Course_name, Course_number, Credit_hours, Department)
VALUES
   ('Intro to Computer Science', 'CS1310', 4, 'CS'),
   ('Data Structures', 'CS3320', 4, 'CS'),
   ('Discrete Mathematics', 'MATH2410', 3, 'MATH'),
   ('Database', 'CS3380', 3, 'CS');

-- Insert data into the Section table
INSERT INTO school.section (Section_identifier, Course_number, Semester, Year, Instructor)
VALUES
   (85, 'MATH2410', 'Fall', 7, 'King'),
   (92, 'CS1310', 'Fall', 7, 'Anderson'),
   (102, 'CS3320', 'Spring', 8, 'Knuth'),
   (112, 'MATH2410', 'Fall', 8, 'Chang'),
   (119, 'CS1310', 'Fall', 8, 'Anderson'),
   (135, 'CS3380', 'Fall', 8, 'Stone');

-- Insert data into the Grade Report table
INSERT INTO school.grade_report (Student_number, Section_identifier, Grade)
VALUES
   (17, 112, 'B'),
   (17, 119, 'C'),
   (8, 85, 'A'),
   (8, 92, 'A'),
   (8, 102, 'B'),
   (8, 135, 'A');

-- Insert data into the Prerequisite table
INSERT INTO school.prerequisite (Course_number, Prerequisite_number)
VALUES
   ('CS3380', 'CS3320'),
   ('CS3380', 'MATH2410'),
   ('CS3320', 'CS1310');
```

--1) Retrieve the course names of all the courses that come under the department of ‘cs’ (computer science).

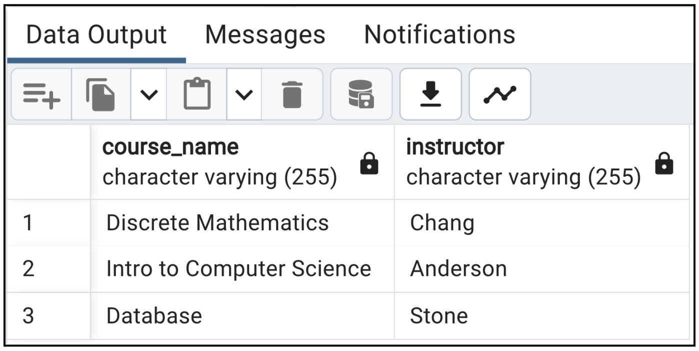
```
SELECT Course_name
FROM school.course
WHERE Department = 'CS';
```



```
   course_name
   character varying (255)
1  Intro to Computer Science
2  Data Structures
3  Database
```

--2) Retrieve the names of all courses along with the name of the instructor taught during the fall of 2008.

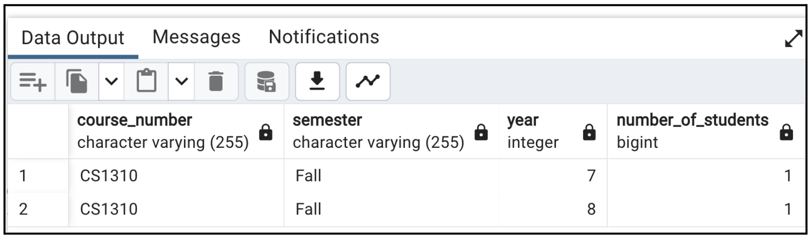
```
SELECT Course.Course_name, Section.Instructor
FROM school.course
JOIN school.section ON Course.Course_number = Section.Course_number
WHERE Section.Semester = 'Fall' AND Section.Year = 8;
```



	course_name character varying (255)	instructor character varying (255)
1	Discrete Mathematics	Chang
2	Intro to Computer Science	Anderson
3	Database	Stone

--3) For each section taught by Professor Anderson, retrieve the course number, semester, year, and number of students who took the section.

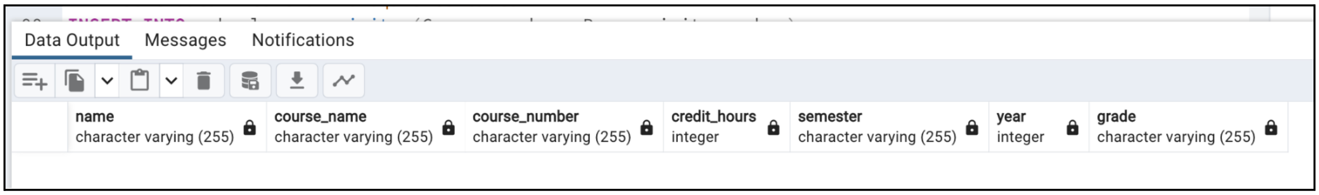
```
SELECT Section.Course_number, Section.Semester, Section.Year, COUNT(Grade_Report.Student_number) AS
Number_of_Students
FROM school.section
JOIN school.grade_report ON Section.Section_identifier = Grade_Report.Section_identifier
JOIN school.student ON Grade_Report.Student_number = Student.Student_number
WHERE Section.Instructor = 'Anderson'
GROUP BY Section.Course_number, Section.Semester, Section.Year;
```



	course_number character varying (255)	semester character varying (255)	year integer	number_of_students bigint
1	CS1310	Fall	7	1
2	CS1310	Fall	8	1

--4) Retrieve the name and transcript of each junior student (Class = 1) majoring in mathematics (MATH). A transcript includes course name, course number, credit hours, semester, year, and grade for each course completed by the student.

```
SELECT Student.Name, Course.Course_name, Course.Course_number, Course.Credit_hours, Section.Semester,
Section.Year, Grade_Report.Grade
FROM school.student
JOIN school.grade_report ON Student.Student_number = Grade_Report.Student_number
JOIN school.section ON Grade_Report.Section_identifier = Section.Section_identifier
JOIN school.course ON Section.Course_number = Course.Course_number
WHERE Student.Class = 1 AND Student.Major = 'MATH';
```



name character varying (255)	course_name character varying (255)	course_number character varying (255)	credit_hours integer	semester character varying (255)	year integer	grade character varying (255)

Exercise 6.13

Write SQL update statements to do the following on the database schema shown in Figure 1.2.

1. Insert a new course, <’Financial Accounting’, ‘fac4390’,5,’BUSINESS’>
2. Insert a new section, <145, ‘fac4390’, ‘Fall’, ‘17’, ‘Hanif’>
3. Insert a new student, <’Robin’, 34, 2, ‘BUSINESS’>.
4. Update the record for the student whose student number is 17 and change his class from 1 to 3.

You only need to submit the SQL for creating the INSERT and UPDATE statements.

--1) Insert a new course, 'Financial Accounting', 'fac4390', 5, 'BUSINESS':

```
INSERT INTO school.course (Course_name, Course_number, Credit_hours, Department)
VALUES ('Financial Accounting', 'fac4390', 5, 'BUSINESS');
```

--2) Insert a new section, 145, 'fac4390', 'Fall', '17', 'Hanif':

```
INSERT INTO school.section (Section_identifier, Course_number, Semester, Year, Instructor)
VALUES (145, 'fac4390', 'Fall', 17, 'Hanif');
```

--3) Insert a new student, 'Robin', 34, 2, 'BUSINESS':

```
INSERT INTO school.student (Name, Student_number, Class, Major)
VALUES ('Robin', 34, 2, 'BUSINESS');
```

--4) Update the record for the student whose student number is 17 and change his class from 1 to 3:

```
UPDATE school.student
SET Class = 3
WHERE Student_number = 17;
```