CSC 411 DBMS Design

Spring 2021

Take-home Midterm Examination

Due Time: 11:59 PM, March 25, 2021

Total points: 100

NAME Jackie Diep

STUDENT ID# W10076331

1. (10 points) Term Definition (2 points for each question).

(a) Database-management system (DBMS)

A collection of interrelated data and a set of programs to access those data.

(b) Logical schema

A schema that describes the database at the logical level.

(c) Transaction

A collection of operations that performs a single logical function in a database application.

(d) Primary key

A candidate key that is chosen by the database designer as the principal means of identifying tuples within a relation.

(e) Referential integrity constraint

A constraint that requires that the values appearing in specified attributes of any tuple in the referencing relation also appear in specified attributes of at least one tuple in the referenced relation.

1. (20 points) SQL (Data Definition Language (DDL) Part).

The follow schema and an instance for the Students relation are given. The primary keys are underlined.

Students (*sid*: integer, *sname*: string, *major*: string)

Takes (*sid*: integer, *cid*: string, *grade*: string)

Courses (*cid*: string, *title*: string)

50000	David	CS
50001	Jones	CS
50002	Smith	Math
50003	Smith	CS
50004	Madayan	CS

(a) [5 points] Write a SQL query to find the student IDs for all students in CS program.

SELECT *sid* **FROM** *Students*
WHERE *major* = 'CS'

(b) [5 points] Write a SQL statement to insert a new record with values (50005, 'Mike', 'CS') to the Students table.

INSERT INTO *Students*
VALUES (50005, 'Mike', 'CS')

(c) [5 points] Write a SQL statement to delete the records of those students who major in Math.

DELETE FROM *Students*
WHERE *major* = 'Math'

DELETE FROM *Takes*
WHERE *sid* **in**
(**SELECT** *sid*
FROM *Students*
WHERE *major* = 'Math')

(d) [5 points] If the student whose *sid* is 50000 and name is David has double majors in both CS and Math. Can we insert a new record with values (50000, 'David', 'Math') to the original Students table? Explain why.

No, there already exists a primary key of 50000, and SQL does not support duplicates of the same primary key due to the primary key constraint.

Depending on the type of **INSERT** command, results may differ; however, the desired outcome will not occur.

2. (25 points) SQL Queries (Data Manipulation Language (DML) Part)

The questions are based on the following relational schema (keys are underlined):

Student (*sid:* integer, s*name*: string, *major*: string, *age*: integer)

Takes (s*id*: integer, *cid*: integer)

Class (*cid*: integer, *cname*: string, *instructor*: string, *room*: string, *meet_at*: time)

(a) [5 points] Write the following query **in SQL**: *Find the names of students who are enrolled in the classes offered by the instructor 'Bo Li'.*

```
SELECT  sname FROM Student
WHERE sid IN
        (SELECT sid FROM Takes
        WHERE cid IN
                (SELECT cid FROM Class
                WHERE instructor = 'Bo Li'
                )
        )
```

(b) [5 points] Write the following query **in SQL**: *Find the names of all students who are enrolled in both 'Database' and 'Algorithm' classes* .

```
SELECT  sname FROM Student
WHERE sid IN
        (SELECT sid FROM Takes
        WHERE cid IN
                (SELECT cid FROM Class
                WHERE cname = 'Database' AND cname = 'Algorithm'
                )
        )
```

(c) [5 points] Write the following query **in SQL**: *Find the names of all students who are enrolled in 'Database' class but not in 'Algorithm' class*.

```
SELECT  sname FROM Student
WHERE sid IN
        (SELECT sid FROM Takes
        WHERE cid IN
                (SELECT cid FROM Class
                WHERE cname = 'Database' AND NOT cname = 'Algorithm'
                )
        )
```

(d) [5 points] Write the following query **in SQL**: *Find the names of all students who are not enrolled in any class.*

SELECT *sname* **FROM** *Student*
WHERE *sid* **NOT IN**
(**SELECT** *sid* **FROM** *Takes*
)

(e) [5 points] Write the following query **in SQL**: *For each class, find the average age of all students who are enrolled in that class.*

SELECT AVG(*age*) **FROM** *Student*
WHERE *sid* **IN**
(**SELECT** *sid* **FROM** *Takes*
GROUP BY *cid*
)

3. (15 points) Relational Algebra and Queries.

Consider the following relational schema (keys are underlined):

Students (*sid*: integer, *sname*: string, *major*: string)

Takes (*sid*: string, *cid*: string, *grade*: string)

Courses (*cid*: string, *title*: string)

(a) [5 points] Write the following query **in relational algebra:** Find the names of the students who major in Math or CS, where 'Math' and 'CS' are the values of the attribute *major* in the *Students* relation.

$\Pi_{sname}\ (\sigma_{major = \text{'Math'} \vee major = \text{'CS'}}\ (Students))$

(b) [5 points] Write the following query **in relational algebra**: Find the names of the students who are enrolled in CSC411 but not in CSC414, where 'CSC411' and 'CSC414' are the values of the attribute *title* in the *Courses* relation.

$\Pi_{sname}\ (\sigma_{title = \text{'CSC411'}}\ (Courses \bowtie Takes \bowtie Students)) - \Pi_{sname}\ (\sigma_{title = \text{'CSC414'}}\ (Courses \bowtie Takes \bowtie Students))$

(c) [5 points] Write the following query **in relational algebra:** Find the names of the students who are enrolled in both 'CSC411' and 'CSC414', where 'CSC411' and 'CSC414' are the values of the attribute *title* in the *Courses* relation.

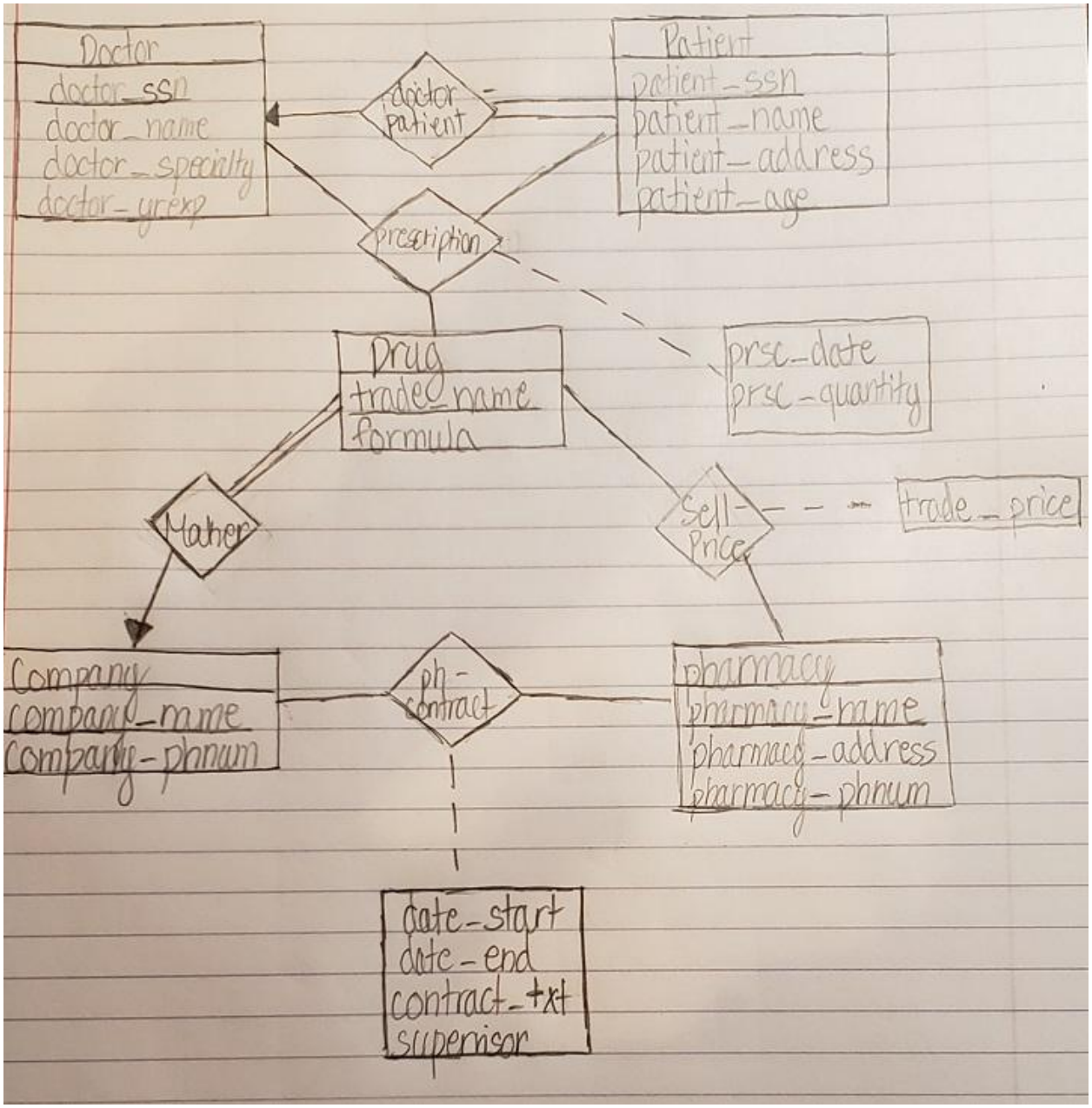
$\Pi_{sname}\ (\sigma_{title\ =\ 'CSC411'\ \wedge\ title\ =\ 'CSC414'}\ (Courses))$

4. (30 points) Database Design and the Entity-Relationship Model

The **Prescriptions-R-X** chain of pharmacies has offered to give you a free lifetime supply of medicine if you design its database. Given the rising cost of health care, you agree. Here's the information that you gather:

- Patients are identified by an SSN, and their names, addresses, and ages must be recorded.
- Doctors are identified by an SSN. For each doctor, the name, specialty, and years of experience must be recorded.
- Each pharmaceutical company is identified by name and has a phone number.
- For each drug, the trade name and formula must be recorded. Each drug is sold by a given pharmaceutical company, and the trade name identifies a drug uniquely from among the products of that company. If a pharmaceutical company is deleted, you need not keep track of its products any longer.
- Each pharmacy has a name, address, and phone number.
- Every patient has a primary physician. Every doctor has at least one patient.
- Each pharmacy sells several drugs and has a price for each. A drug could be sold at several pharmacies, and the price could vary from one pharmacy to another.
- Doctors prescribe drugs for patients. A doctor could prescribe one or more drugs for several patients, and a patient could obtain prescriptions from several doctors. Each prescription has a date and a quantity associated with it. You can assume that, if a doctor prescribes the same drug for the same patient more than once, only the last such prescription needs to be stored.
- Pharmaceutical companies have long-term contracts with pharmacies. A pharmaceutical company can contract with several pharmacies, and a pharmacy can contract with several pharmaceutical companies. For each contract, you have to store a start date, an end date, and the text of the contract.
- Pharmacies appoint a supervisor for each contract. There must always be a supervisor for each contract, but the contract supervisor can change over the lifetime of the contract.

6.1 (20 points) Draw the Entity-Relationship diagram for the above application, including entities, attributes, and underlined primary keys.


Doctor
doctor_ssn
doctor_name
doctor_specialty
doctor_yrexp
doctor patient
Patient
patient_ssn
patient_name
patient_address
patient_age
prescription
prsc_date
prsc_quantity
Drug
trade_name
formula
Maker
Sell-Price
trade_price
Company
company_name
company_phnum
ph-contract
pharmacy
pharmacy_name
pharmacy_address
pharmacy_phnum
date_start
date_end
contract_txt
supervisor

6.1 (10 points) Construct appropriate relation schemas for the E-R diagram created in 6.1. Please also include a list of constraints including primary-key and foreign-key constraints.

doctor(doctor_ssn, doctor_name, doctor_specialty, doctor_yrexp)

patient(patient_ssn, doctor_ssn, patient_name, patient_address, patient_age)

foreign key *doctor_ssn* **references** *doctor*

doctor-patient(patient_ssn, doctor_ssn)

foreign key *patient_ssn* **references** *patient*

foreign key *doctor_ssn* **references** *doctor*

company(company_name, company_phnum)

drug(trade_name, company_name, formula)

foreign key *company_name* **references** *company*

pharmacy(pharmacy_name, pharmacy_address, pharmacy_phnum)

sell-price(pharmacy_name, trade_name, trade_price)

foreign key *pharmacy_name* **references** *pharmacy*

foreign key *trade_name* **references** *drug*

prescription(doctor_ssn, patient_ssn, trade_name, prsc_date, prsc_quantity)

foreign key *doctor_ssn* **references** *doctor*

foreign key *patient_ssn* **references** *patient*

foreign key *trade_name* **references** *drug*

ph-contract(company_name, pharmacy_name, date_start, date_end, contract_txt, supervisor)

foreign key *company_name* **references** *company*

foreign key *pharmacy_name* **references** *pharmacy*