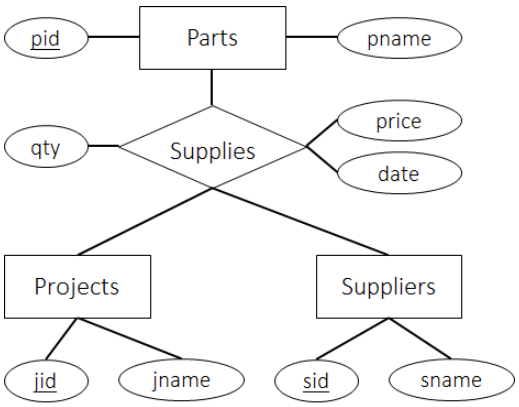
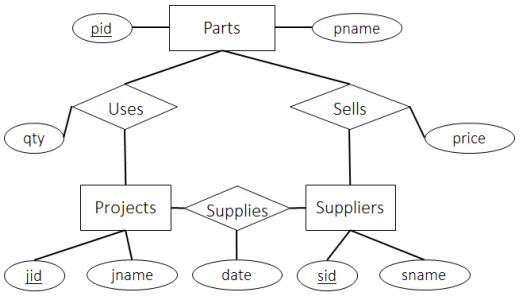
1. Consider an application about the parts (*with identifier* ***pid*** *and name* ***pname***) supplied by suppliers (*with identifier* ***sid*** *and name* ***sname***) to projects (*with identifier* ***jid*** *and name* ***jname***) where `price` represents the unit price of a part, `qty` represents the quantity of a part and `date` represents the date of transaction.

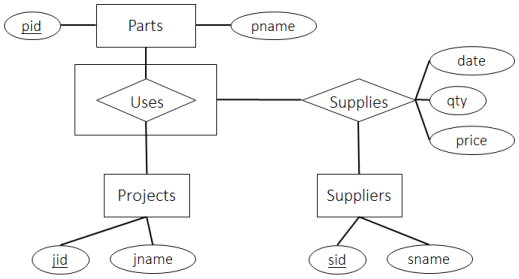
   The following are three different ER designs for the application



(a) Design A



(b) Design B



(c) Design C

   (a) Discuss whether the designs are *equivalent* in the sense of capturing the same constraints of the application.

   (b) Translate each of the ER diagram into a relational schema. Assume appropriate domains for the attributes.

Solution:

(a) In Design A, if (S, P, J) is in `Supplies` relationship set, the supplier S supplies part P to project J. Moreover, the value of each of the relationship attributes `date`, `qty` and `price` depends on the (S, P, J) triple. For example, supplier S could be selling part P to project J for \$10 each BUT the same supplier S could be selling the same part P to another project J' for \$15 each.

When the ternary relationship in Design A is replaced by a collection of binary relationship in Design B, the semantics are not the same. First, each of the value of the relationship attributes `date`, `qty` and `price` in Design B depends only on a pair of entities. For example, the price of a part P sold by supplier S is fixed for *all projects*.

Second, even if:

- (S, P) is in `Sells` relationship set.
  - Supplier S sells part P.
- (S, J) is in `Supplies` relationship set.
  - Supplier S supplies to project J.
- (J, P) is in `Uses` relationship set.
  - Project J uses part P.

The three facts do **NOT** necessarily mean that supplier S sells part P to project J.

Design C is more *general* than Design A as the former (*i.e.*, Design C) has an additional `Uses` relationship set that captures the parts that are used by projects independent of whether a part is being supplied by any supplier.

⚠ For the described application, Design A is sufficient.

(b) Design A

```sql
DROP TABLE IF EXISTS
  Parts, Projects, Suppliers, Supplies CASCADE;

CREATE TABLE Parts (
  pid    INTEGER PRIMARY KEY,
  pname  TEXT
);

CREATE TABLE Projects (
  jid    INTEGER PRIMARY KEY,
  jname  TEXT
);

CREATE TABLE Suppliers (
  sid    INTEGER PRIMARY KEY,
  sname  TEXT
);

CREATE TABLE Supplies (
  pid     INTEGER REFERENCES Parts,
  jid     INTEGER REFERENCES Projects,
  sid     INTEGER REFERENCES Suppliers,
  price   NUMERIC,
  qty     INTEGER,
  t_date  DATE,
  PRIMARY KEY (pid, jid, sid)
);
```

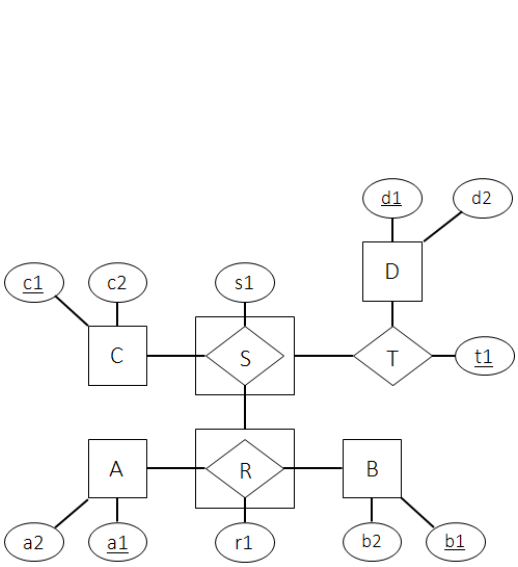
Design B

```
DROP TABLE IF EXISTS
  Parts, Projects, Suppliers, Uses, Sells, Supplies
CASCADE;

CREATE TABLE Parts (
  pid     INTEGER PRIMARY KEY,
  pname   TEXT
);

CREATE TABLE Projects (
  jid     INTEGER PRIMARY KEY,
  jname   TEXT
);

CREATE TABLE Suppliers (
  sid     INTEGER PRIMARY KEY,
  sname   TEXT
);

CREATE TABLE Uses (
  pid      INTEGER REFERENCES Parts,
  jid      INTEGER REFERENCES Projects,
  qty      INTEGER,
  PRIMARY KEY (pid, jid)
);

CREATE TABLE Sells (
  pid      INTEGER REFERENCES Parts,
  sid      INTEGER REFERENCES Projects,
  price    NUMERIC,
  PRIMARY KEY (pid, sid)
);

CREATE TABLE Supplies (
  jid      INTEGER REFERENCES Projects,
  sid      INTEGER REFERENCES Suppliers,
  t_date   DATE,
  PRIMARY KEY (jid, sid)
);
```

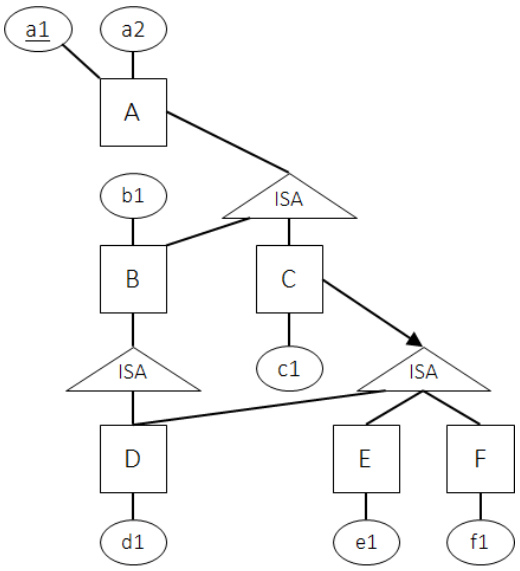
Design C

```
DROP TABLE IF EXISTS
  Parts, Projects, Suppliers, Uses, Supplies
CASCADE;

CREATE TABLE Parts (
  pid     INTEGER PRIMARY KEY,
  pname   TEXT
);

CREATE TABLE Projects (
  jid     INTEGER PRIMARY KEY,
  jname   TEXT
);

CREATE TABLE Suppliers (
  sid     INTEGER PRIMARY KEY,
  sname   TEXT
);

CREATE TABLE Uses (
  pid      INTEGER REFERENCES Parts,
  jid      INTEGER REFERENCES Projects,
  PRIMARY KEY (pid, jid)
);

CREATE TABLE Supplies (
  pid      INTEGER,
  jid      INTEGER,
  sid      INTEGER REFERENCES Suppliers,
  price    NUMERIC,
  qty      INTEGER,
  t_date   DATE,
  PRIMARY KEY (pid, jid, sid),
  FOREIGN KEY (pid, jid) REFERENCES Uses
);
```

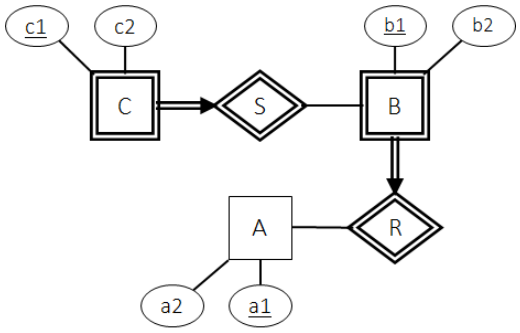
2. For each of the ER diagrams shown below, translate it into a relational schema. You may assume that all the attributes have integer domain.



(a) ER 1



(b) ER 2



(c) ER 3

Solution:

(a) ER 1

```sql
DROP TABLE IF EXISTS A, B, C, D, R, S, T CASCADE;

CREATE TABLE A (
  a1  INTEGER PRIMARY KEY,
  a2  INTEGER
);

CREATE TABLE B (
  b1  INTEGER PRIMARY KEY,
  b2  INTEGER
);

CREATE TABLE C (
  c1  INTEGER PRIMARY KEY,
  c2  INTEGER
);
```

```
17
18 CREATE TABLE D (
19   d1  INTEGER PRIMARY KEY,
20   d2  INTEGER
21 );
22
23 CREATE TABLE R (
24   a1  INTEGER REFERENCES A,
25   b1  INTEGER REFERENCES B,
26   r1  INTEGER,
27   PRIMARY KEY (a1, b1)
28 );
29
30 CREATE TABLE S (
31   a1  INTEGER,
32   b1  INTEGER,
33   c1  INTEGER REFERENCES C,
34   s1  INTEGER,
35   PRIMARY KEY (a1, b1, c1),
36   FOREIGN KEY (a1, b1) REFERENCES R
37 );
38
39 CREATE TABLE T (
40   a1  INTEGER,
41   b1  INTEGER,
42   c1  INTEGER,
43   d1  INTEGER REFERENCES D,
44   t1  INTEGER,
45   PRIMARY KEY (a1, b1, c1, d1, t1),
46   FOREIGN KEY (a1, b1, c1) REFERENCES S
47 );
```

(b) ER 2

```
1  DROP TABLE IF EXISTS A, B, C, D, E, F CASCADE;
2
3  CREATE TABLE A (
4    a1  INTEGER PRIMARY KEY,
5    a2  INTEGER
6  );
7
8  CREATE TABLE B (
9    a1  INTEGER PRIMARY KEY REFERENCES A
10         ON DELETE CASCADE,
11   b1  INTEGER
12 );
13
14
15 CREATE TABLE C (
16   a1  INTEGER PRIMARY KEY REFERENCES A
17         ON DELETE CASCADE,
18   c1  INTEGER
19 );
20
21 CREATE TABLE D (
22   a1  INTEGER PRIMARY KEY REFERENCES B REFERENCES C
23         ON DELETE CASCADE,
```

```
  d1  INTEGER
);

CREATE TABLE E (
  a1  INTEGER PRIMARY KEY REFERENCES C
        ON DELETE CASCADE,
  e1  INTEGER
);

CREATE TABLE F (
  a1  INTEGER PRIMARY KEY REFERENCES C
        ON DELETE CASCADE,
  f1  INTEGER
);
```

(c) ER 3

```
DROP TABLE IF EXISTS A, B, C CASCADE;

CREATE TABLE A (
  a1  INTEGER PRIMARY KEY,
  a2  INTEGER
);

CREATE TABLE B (
  a1  INTEGER REFERENCES A
        ON DELETE CASCADE,
  b1  INTEGER,
  b2  INTEGER,
  PRIMARY KEY (a1, b1)
);

CREATE TABLE C (
  a1  INTEGER,
  b1  INTEGER,
  c1  INTEGER,
  c2  INTEGER,
  PRIMARY KEY (a1, b1, c1),
  FOREIGN KEY (a1, b1) REFERENCES B
        ON DELETE CASCADE
);
```

3. The Prescription-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 healt care, you agree. Here's the information that you gather:

    1. Patients are identified by an SSN. Their names, addresses and ages must also be recorded.
    2. Doctors are identified by an SSN. Their name, specialty and years of experience must also be recorded.
    3. Each pharmaceutical company is identified by name and has a phone number.
    4. For each drug, the *trade name* and formula must be recorded.
    5. 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.
    6. Each pharmacy has a name, address and phone number.
    7. Every patient has a primary physician.
    8. Every doctor is the primary physician of at least one patient.
    9. 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.
    10. 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.
    11. There is exactly one contract between a pharmacy and a pharmaceutical company if and only if that pharmacy sells some drug that is made by that pharmaceutical company. For each contract, you have to store a start date, an end date and the text of the contract.

    Questions:

    (a) Consider the ER diagram shown on the next page for Prescriptions-R-X. What are the constraints that are not captured by this design? Modify the ER design to capture as many of the constraints as possible.

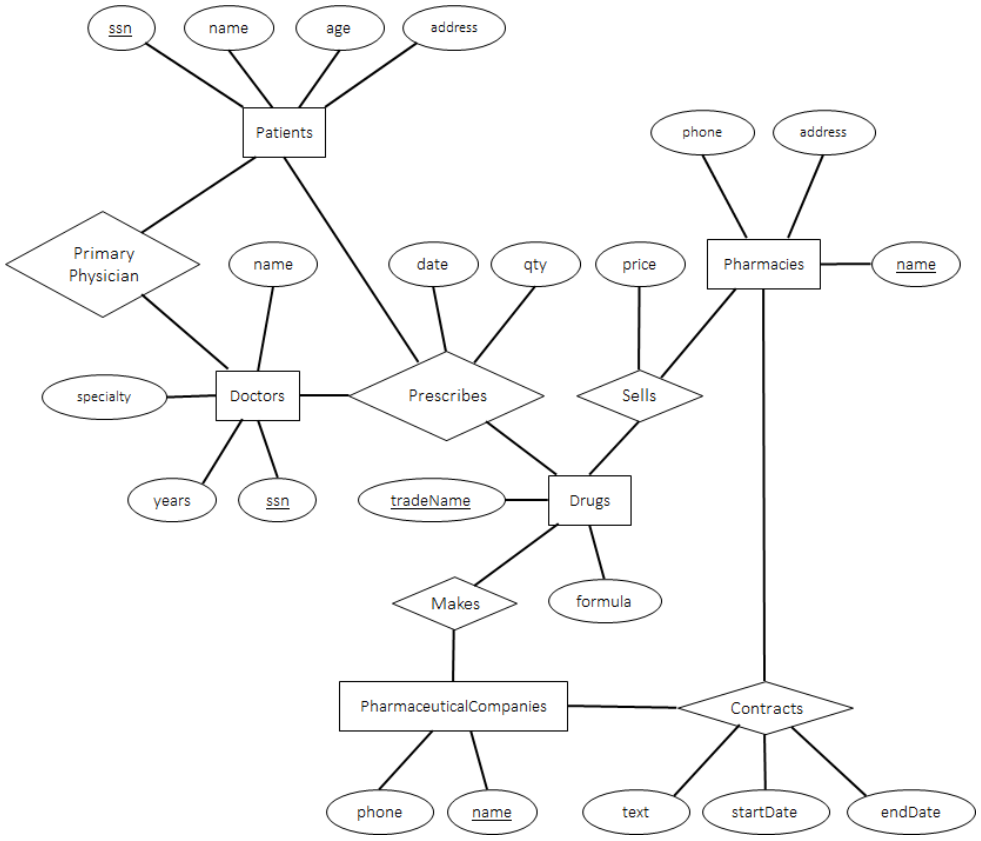
    (b) Translate your ER design in part (a) into a relational schema using SQL (*assume reasonable data types for the domain constraints*). Your solution should capture as many of the application's constraints as possible. Identify any constraints that are not captured by your relational schema.

    (c) How would your design in part (a) change if each drug must be sold at a fixed price by all pharmacies?

    (d) How would your design in part (a) change if the design requirements change as follows: "*if a doctor prescribes the same drug for the same patient more than once, several such prescriptions may have to be stored*".

(e) Suppose that 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. Supervisors are identified by an SSN and their start dates must be recorded. Modify your ER design in part (a) to capture this additional requirement.

(f) Translate your ER design in part (e) into a relational schema. Identify any constraints that are not captured by your relational schema.



Solution:

(a) The following constraints are not captured by the ER design:

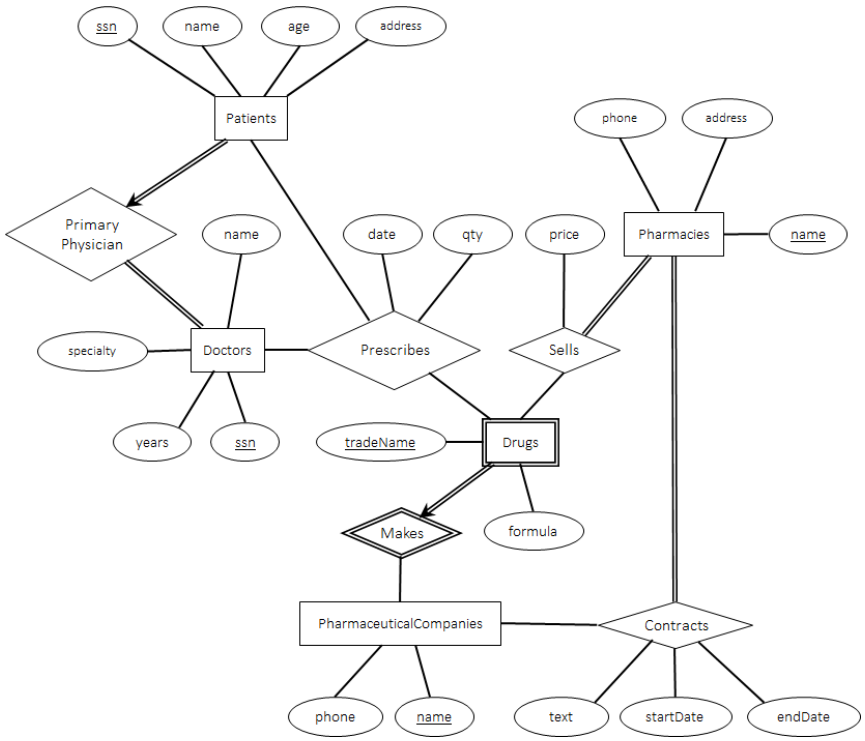
C1. `Drugs` is a weak entity set that is dependent on the owner entity set `PharmaceuticalCompanies`.

C2. The key and total participation constraints on `Patients` w.r.t. `PrimaryPhysician` relationship.

C3. The total participation constraint on `Doctors` w.r.t. `Primary-Physician` relationship.

C4. The constraint that each pharmacy sells more than one drug.

C5. The constraint that there is exactly one contract between a pharmaceutical company and a pharmacy if and only if that pharmacy sells some drug that is made by that pharmaceutical company.



The revised ER design shown above captures constraints C1, C2 and C3. Constraint C4 is captured only partially (*total participation of* ***`Pharmacies`*** *w.r.t.* ***`Sells`*** *relationship*) but it does not require that each pharmacy sells more than one drug.

Constraint C5 is not fully captured as the ER design merely specifies that every pharmacy P must have at least one contract with some pharmaceutical company PC but the ER design does not require that P must sell some drug that is made by PC.

(b) Relational schema

```
DROP TABLE IF EXISTS
  Doctors, Patients, Pharmacies, Drugs, Sells,
  PharmaceuticalCompanies, Prescribes, Contracts
CASCADE;

CREATE TABLE Doctors (
  ssn         TEXT PRIMARY KEY,
  name        TEXT,
  specialty   TEXT,
  years       INTEGER
);

CREATE TABLE Patients (
  ssn         TEXT PRIMARY KEY,
  physician   TEXT NOT NULL REFERENCES Doctors,
  name        TEXT,
  address     TEXT,
  age         INTEGER
);
CREATE TABLE Pharmacies (
```

```
  name        TEXT PRIMARY KEY,
  phone       TEXT,
  address     TEXT,
);

CREATE TABLE PharmaceuticalCompanies (
  name        TEXT PRIMARY KEY,
  phone       TEXT
);

CREATE TABLE Drugs (
  pcname      TEXT REFERENCES PharmaceuticalCompanies
                ON DELETE CASCADE,
  tradename   TEXT,
  formula     TEXT,
  PRIMARY KEY (pcname, tradename)
);

CREATE TABLE Prescribes (
  dssn        TEXT REFERENCES Doctors,
  pssn        TEXT REFERENCES Patients,
  pcname      TEXT REFERENCES PharmaceuticalCompanies
                ON DELETE CASCADE,
  tradename   TEXT,
  pdate       DATE,
  qty         INTEGER,
  PRIMARY KEY (dssn, pssn, pcname, tradename),
  FOREIGN KEY (pcname, tradename) REFERENCES Drugs
);

CREATE TABLE Contracts (
  pname       TEXT REFERENCES Pharmacies,
  pcname      TEXT REFERENCES PharmaceuticalCompanies
                ON DELETE CASCADE,
  start_date  DATE,
  end_date    DATE,
  comments    TEXT,
  PRIMARY KEY (pname, pcname)
);

CREATE TABLE Sells (
  pname       TEXT REFERENCES Pharmacies,
  pcname      TEXT REFERENCES PharmaceuticalCompanies
                ON DELETE CASCADE,
  tradename   TEXT,
  price       NUMERIC,
  PRIMARY KEY (pname, pcname, tradename),
  FOREIGN KEY (pname, pcname) REFERENCES Contracts,
  FOREIGN KEY (pcname, tradename) REFERENCES Drugs
);
```

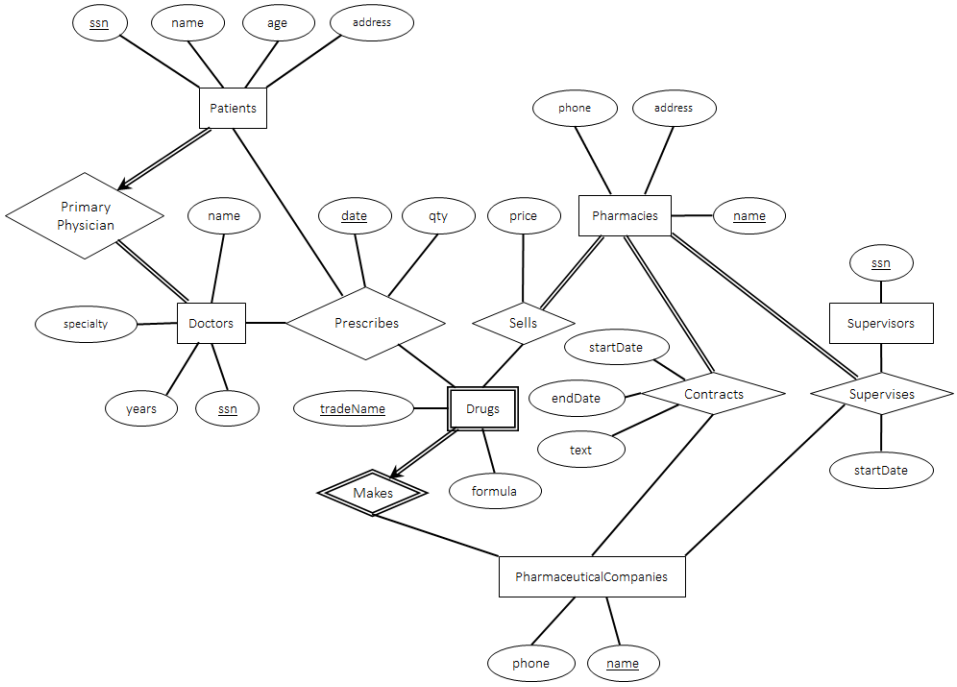
The above relational schema does not enforce constraints C3 and C4. Constraint C5 is enforced only partially: the foreign key constraint from `Sells` to `Contracts` guarantees that if a pharmacy sells some drug made by a pharmaceutical company, then there is a contract between that pharmacy and pharmaceutical company. However, it is

possible for a contract to exist in the database between a pharmacy and a pharmaceutical company where that pharmacy does not sell any drug made by that pharmaceutical company.

(c) Instead of modeling `price` as an attribute of `Sells` relationship set, we model `price` as an attribute of `Drugs` entity set.

(d) The `date` attribute of `Prescribes` is changed to a key attribute. Thus, `Key(Prescribes)` consists of the following attributes: `Key(Patients)`, `Key(Drugs)` and `date`, where `Key(Patients) = {ssn}` and `Key(Drugs) = {name, tradename}`.

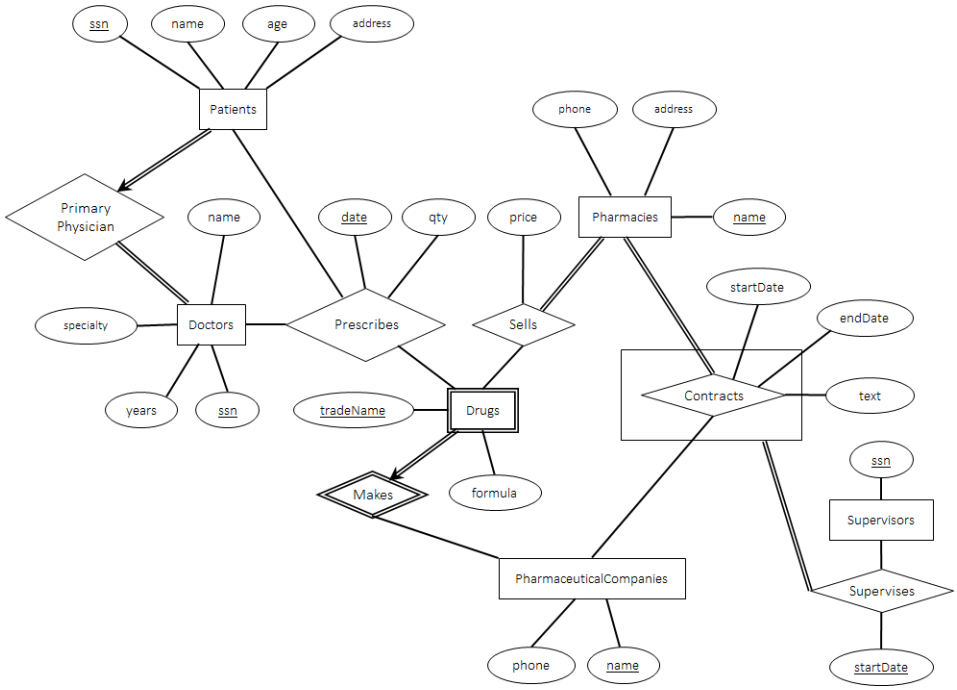
(e) Consider the following ER design which introduces a new *ternary* relationship set "`Supervises`" to related `Pharmacies`, `PharmaceuticalCompanies` and a new entity set "`Supervisors`". Note that if a pharmacy P and a supervisor S is related by a `Supervises` relationship, then it is implicitly means that S is appointed by P. Since each pharmacy has at least one contract, a total participation constrant on `Pharmacies` w.r.t. `Supervises` is required.



The revised ER design does not completely capture the new requirement: the existence of (P, PC, S) in `Supervises` relationship does not necessarily mean that there exist a contract between pharmacy P and a pharmaceutical company PC. Furthermore, this Er design also does not capture the constraint that each contract is supervised by only one contractor at any time. Finally, this ER design also does not allow a supervisor to supervise the same contract multiple times (*over a different time periods*).

An alternative ER design (*shown below*) is to model `Supervises` as a binary relationship between `Contract` (*as an* ***aggregation***) and

`Supervisors`. Also, the attribute `start_date` (*`startDate` in the ER design*) is changed to be part of the key of `Supervises`. This ensures that each contract will be supervised by some supervisor and a supervisor could supervise the same contract multiple times.



Similar to the previous design, this design does not capture the constraint that each contract should be supervised by only one supervisor at any time.

(f) The following shows the additional relational tables derived from the last aggregation-based ER diagram.

```
1  CREATE TABLE Supervisors (
2    ssn         TEXT PRIMARY KEY
3  );
4
5  CREATE TABLE Supervises (
6    pcname      TEXT,
7    pname       TEXT,
8    ssn         TEXT NOT NULL REFERENCES Supervisors,
9    start_date  DATE,
10   PRIMARY KEY (pname, pcname, start_date),
11   FOREIGN KEY (pname, pcname) REFERENCES Contracts
12 );
```

The primary key of `Supervises` ensures that there can't be more than one supervisor supervising a contract at the same time. The foreign key constraint from `Supervises` to `Contracts` ensures that every supervised pair of pharmacy and pharmaceutical company indeed has a contract between them. However, the above schema does not enforce the total participation constraint of `Contracts` w.r.t. `Supervises`.