CS2102 Tutorial 3

1. a) **Design A**

(S,P,J) in Supplies relationship set) means Supplier S supplies part P to project J

value of attributes (qty, price and date) depend on the (S,P,J) tuples

e.g S sell P to J for $10 } same S,P but different price to different J
but S sell P to J2 for $15

Design B

value of attributes (date, qty, price) depends on pairs of entities

e.g price of P sold by S is fixed for all projects
qty of P used by J is same regardless of supplier/price
date of S supplying to J is fixed

Also, (S,P) in sells
(S,J) in supplies } $\neq$ S sells P to J
(J,P) in uses

Design C

J uses P is treated as a higher level entity set — project uses parts regardless of who supplies

S supplies to a J using P

(qty, price and date) is fixed for each S supplying to a (J using P)

b) **Design A**

many to Many ternary relationship set

```
CREATE TABLE Supplies (
    pid   INTEGER
    jid   INTEGER
    sid   INTEGER
    qty   INTEGER
    price INTEGER
    date  DATE

    PRIMARY KEY ( pid, jid, sid )
    FOREIGN KEY ( pid ) REFERENCES  Parts (pid)
         "      ( jid )     "       Project (jid)
         "      ( sid )     "       Supplies (sid)

);
```

Design B

Many to Many pairwise relationship set

```
CREATE TABLE Uses (
    pid  INTEGER
    jid  INTEGER
    qty  INTEGER
    PK (pid, jid)
    FK (pid) REF Parts (pid)
    FK (jid) REF Projects (jid)
);
```

```
CREATE TABLE Supplier (
    jid   INTEGER
    sid   INTEGER
    date  DATE
    PK    (sid, jid)
    FK    (sid) REF supplies (sid)
    FK    (jid) REF Projects (jid)
);
```

```
CREATE TABLE Sells (
    pid   INTEGER
    sid   INTEGER
    price INTEGER
    PK  (pid, sid)
    FK  (pid) REF Parts (pid)
    FK  (sid) REF Supplies (sid)
);
```

DESIGN C

Aggregation Many to Many

```
CREATE TABLE Supplies (
    pid   INTEGER
    jid   INTEGER
    sid   INTEGER
    date  DATE
    qty   INTEGER
    price INTEGER
    PK ( pid, jid, sid )
    FK ( sid ) REF suppliers (sid)
    FK ( pid, jid ) REF Uses ( pid, jid )
);
```

2. a) ER 1

C entity to $\boxed{R}$ entity only   ignore T ◇ first

```
CREATE TABLE S (
    c1 INT
    a1 INT
    b1 INT
    s1 INT
    PK ( a1, b1, c1 )
    FK ( c1 ) REF C (c1)
    FK ( a1, b1, c1 ) REF R ( a1, b1, c1 )
);
```

```
CREATE TABLE R (
    a1 INT
    b1 INT
    c1 INT
    r1 INT
    PK ( a1, b1, c1 )
    FK ( c1 ) REF S ( c1, a1, b1 )
    FK ( a1 ) REF A(a1)
    FK ( b1 ) REF B ( b1 )
);
```

```
CREATE TABLE T (
    a1 INT
    b1 INT
    c1 INT
    d1 INT
    t1 INT
    PK ( a1, b1, c1, d1 )
    FK ( d1 ) REF D( d1 )
    FK ( a1, b1, c1 ) REF C
);
```

b) ER2

A can be both B and C
A can be nether B and C

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

```
CREATE TABLE B (
    a1  INTEGER  PK
    b1  INTEGER
    FK (a1) REF A (a1) ONDELETE
                       CASCADE
);
```

```
CREATE TABLE C (
    a1  INT   PK
    c1  INT
    FK (a1)  REF(a1) ODC
);
```

```
CREATE TABLE D (
    a1  INT  PK
    d1  INT
    FK (a1) REF B (a1) ODC
    FK (a1) REF C (a2) ODC
);
```

C can be nether D,E,F
C cannot be more than D,E or F

```
CREATE TABLE E (
    a1  INT
    e1  INT
    FK (a1) REF C (a1) ODC
);
```

```
CREATE TABLE F (
    a1
    f1
    FK (a1) REF C (a1) ODC
);
```

c) ER3

```
CREATE TABLE A (
    a1  INT  PK
    a2  INT
);
```

```
CREATE TABLE B (
    a1  INT
    b1  INT
    b2  INT
    PK (a1,b2)
    FK (a1)  REF  A(a1)  ODC
);
```

```
CREATE TABLE C (
    a1  INT
    b1  INT
    c1  INT
    c2  INT
    PK (a1, b2, c2)
    FK (a1,b1) ref  B (a1,b1)  ODC
);
```

3.

a)

5. Drug is a weak entity set dependent on owner entity set Pharmaceutical Companies

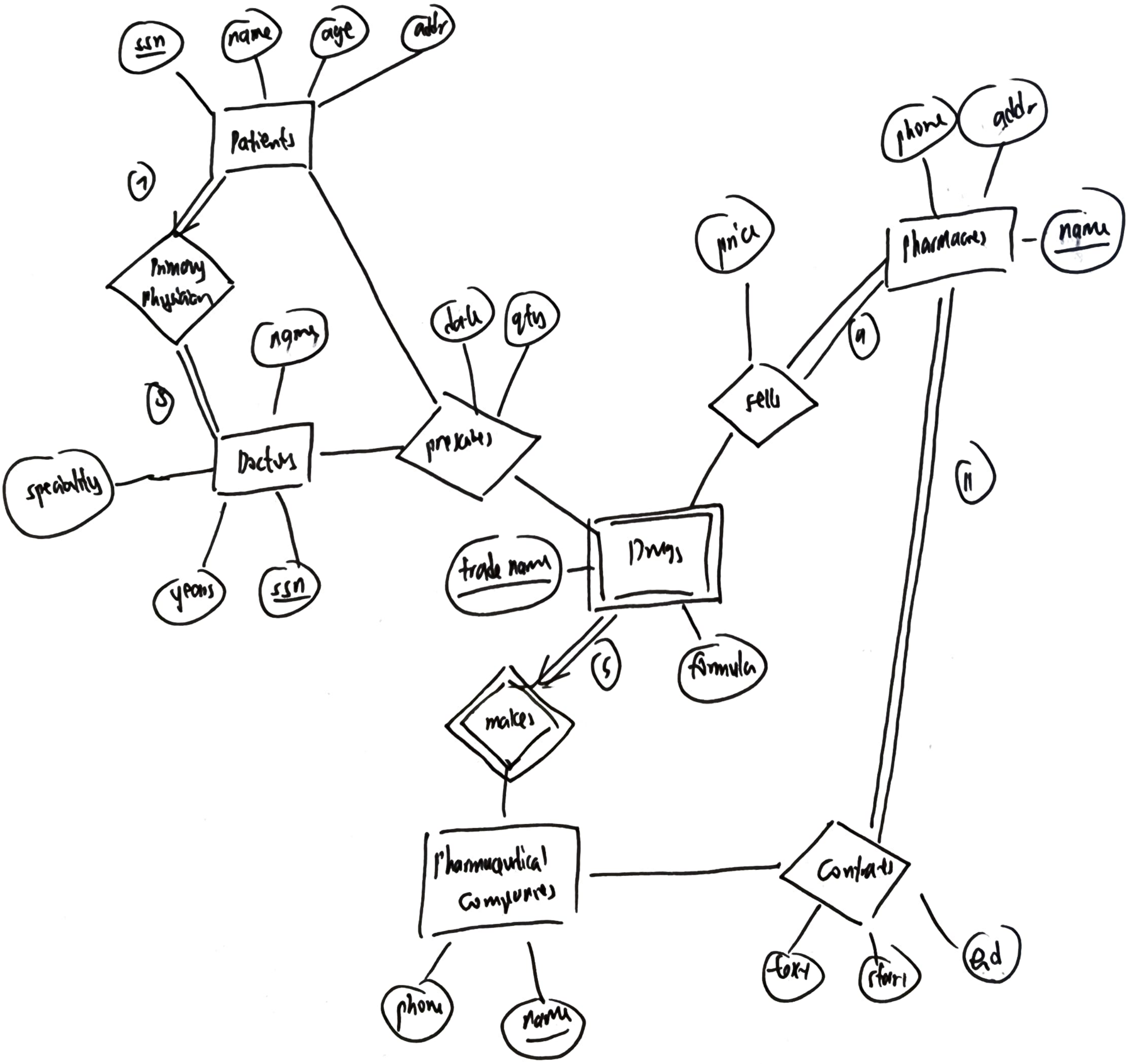
7. key and total participation (exactly 1) of Patient, wrt Primary Physician relationship

8. total participation of Doctors wrt PrimaryPhysician relationship

9. each pharmacy sells more than 1 drug (only total participation 1..* does not enforce >1)

10. Doctor, Drug, Patient relationship

11. exactly 1 contract between Pharmacy and Company iff sell common drug (not fully captured)



h)

```
CREATE TABLE Doctors (

  ssn     INT           PK
  name    VARCHAR (50)
  spec    VARCHAR (50)
  years   INT

);
```

& PP to 1 [illegible]

```
CREATE TABLE Patients (

  ssn   INT   PK
  name  VC (50)
  addr  VC (50)
  age   INT
  PP    INT   NOT NULL

  FK (PP) REF Doctors (ssn)

);
```

```
CREATE TABLE Pharmacies (

  name   VC (50)  PK
  phone  VC (50)
  addr   VC (50)

);
```

```
CREATE TABLE PC (

  name   VC (50)   PK
  phone  VC (50)

);
```

```
CREATE TABLE Sells (

  pname   VC(50)
  pcname  VC(50)
  tname   VC(50)
  price   INT

  PK ( pname, pcname, tname )
  FK (pname, pcname) REF contracts( pname, pcname)
  FK (pcname, tname) REF Drugs (pname, pcname)

);
```

weak entity rel to PC

```
CREATE TABLE Drugs (

  name       VC (50)
  tradename  VC (50)
  formula    VC (50)

  PK ( name, tradename)

  FK ( name)   REF   PC (name)   ODC

);
```

FK from sells to contracts guarantees if P sells drug made by PC, there is a contract

```
CREATE TABLE Contracts (

  pcname  VC (50)
  pname   VC (50)
  start   DATE
  end     DATE
  text    VC (50)

  PK ( pname, pcname)
  FK (pcname) REF PC (name)
  FK (pname) REF Pharmacies (name)

);
```

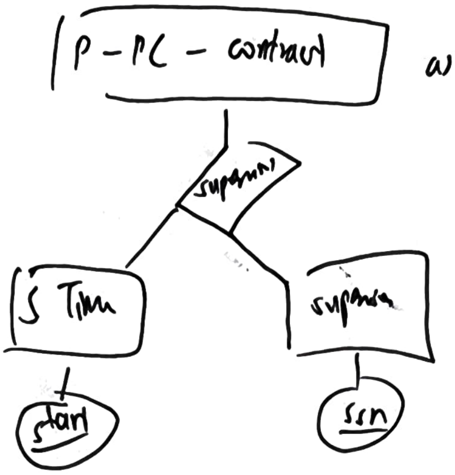
```
CREATE TABLE Prescribes (

  p ssn   INT
  d ssn   INT
  name    VC (50)
  tname   VC (50)
  qty     INT
  date    DATE

  PK( pssn, dssn, name, tname)
  FK (pssn) REF Patients (ssn)
  FK (dssn) REF Doctors (ssn)
  FK (name, tname) REF Drugs ( name, tname)

);
```

contract can exist w/o P & PC sell but not vice versa

c) Drug sold at fixed price => make price an attribute of Drugs itself

d) model as a 4-ary relationship with additional entity set Visits with PK dates

e) treat P-PC-contract as a higher entity set



supervisor can supervise same contract multiple times

f)

```
CREATE TABLE Supervisor (
    ssn  VC(50) PK
);
```

```
CREATE TABLE (SupervisorTime(
    start DATE PK
);
```

```
CREATE TABLE Supervises (
    pcname VC
    pname VC
    ssn  VC      NOT NULL
    start DATE

    PK ( pname, pcname, start)
    FK ( ssn) REF  supervisor( ssn )
    FK ( start) REF  STime(start)
    FK ( pname, pcname) REF  Contracts ( pname, pcname)

);
```