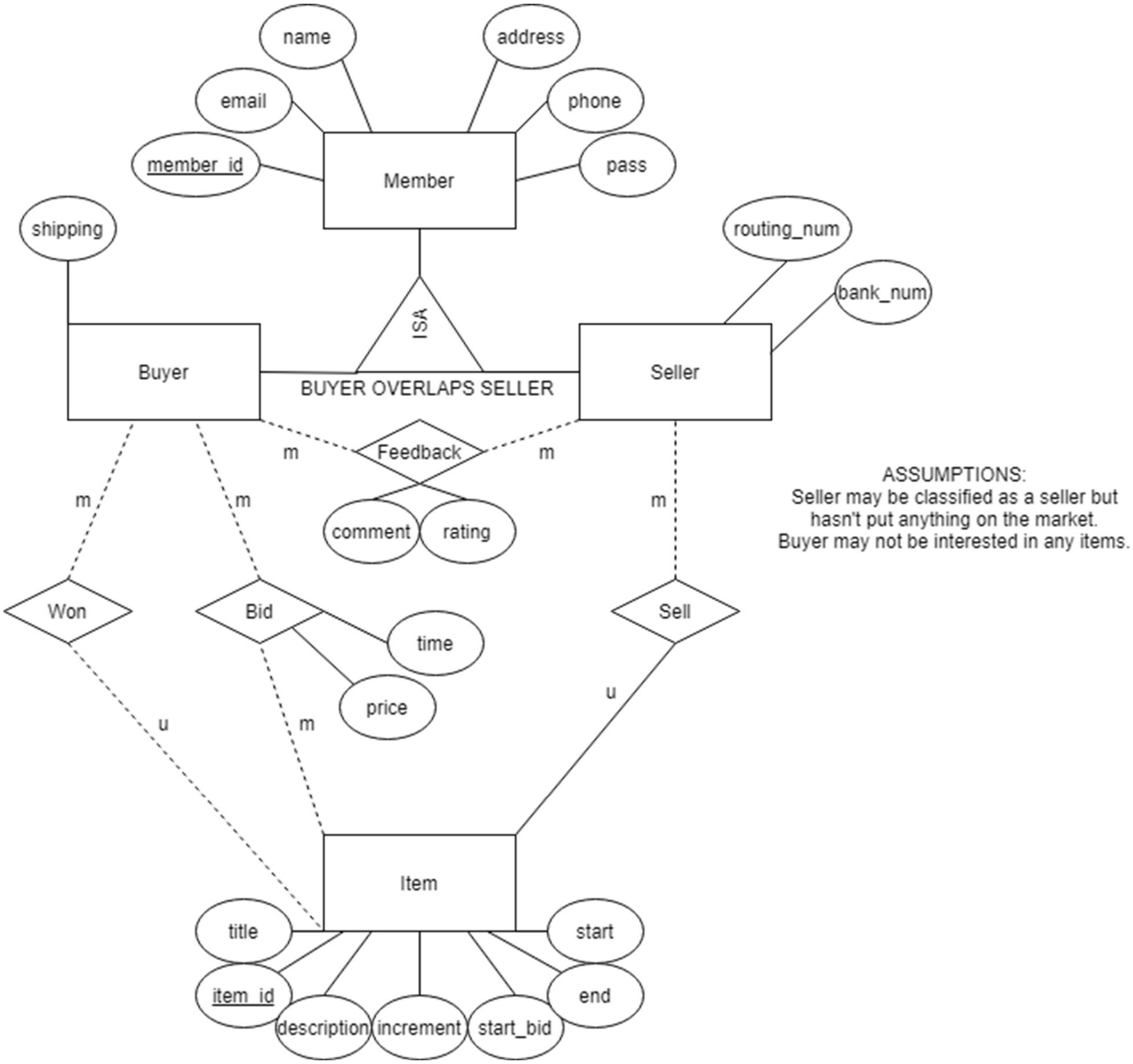


ASSUMPTIONS:
Seller may be classified as a seller but hasn't put anything on the market.
Buyer may not be interested in any items.

1.
2. SQL

```
CREATE TABLE IF NOT EXISTS faculty(
        FID int,
   address varchar(80),
   phone int,
   name varchar(42),
   PRIMARY KEY(FID)
);
```

```
CREATE TABLE IF NOT EXISTS students(
        SID int,
   name varchar(42),
   degree varchar(20),
   advisor_ID int NOT NULL, # total-uni particpation "workaround"
   FOREIGN KEY(advisor_ID) REFERENCES faculty(FID),
        PRIMARY KEY(SID)
);
CREATE TABLE IF NOT EXISTS labOffice(
        ID int,
   seats int,
   address varchar(50),
   PRIMARY KEY(ID)
);
CREATE TABLE IF NOT EXISTS advises(
        FID int,
   SID int UNIQUE, # gurantees only one advisor relationship per student
   PRIMARY KEY(SID),
   FOREIGN KEY(FID) REFERENCES faculty(FID),
   FOREIGN KEY(SID) REFERENCES students(SID)
);
CREATE TABLE IF NOT EXISTS works(
        office_id int,
   sid int,
   since datetime,
        FOREIGN KEY(office_id) REFERENCES labOffice(ID),
   FOREIGN KEY(sid) REFERENCES students(SID)
)
```

3. Relational Algebra

a.

Level
Undergraduate

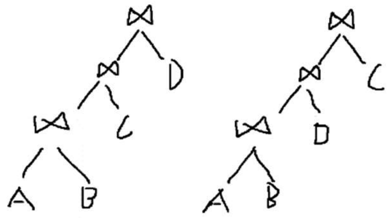
b.

Students.snum	**Students.name**	**gender**	**Majors.snum**	**Majors.name**	**level**
1001	Randy	M	1001	Computer Science	BS
1001	Randy	M	1005	Applied Mathematics	MS
1005	Nicole	F	1001	Computer Science	BS
1005	Nicole	F	1005	Applied Mathematics	MS

c.

snum	**Minors.name**	**Minors.level**	**Degrees.name**	**Degrees.level**	**department_code**
1005	Computer Science	BS	Computer Science	BS	401
1005	Computer Science	BS	Computer Science	MS	401
1005	Computer Science	BS	Computer Science	PHD	401
1001	Software Engineering	BS	Software Engineering	BS	401

d. SELECT m.name FROM Students s JOIN Majors m ON s.snum=m.snum AND s.name = “Randy”;

4. Left-Deep
    a. Allows us to generate fully pipelined plans while reducing the search space
    b. 

5. Block Nested Loop Join
    a. M*Cr seconds to load R
    b. N*Cr seconds to load S
    c. (M+ceil(M/(B-2))*N)*Cw seconds to write results
6. Sorting
    a. Pass 1: Load 3 pages at a time. Sort and merge them. 30/3 = 10 sorted lists
    b. Pass 2: Load 2 lists in 2 pages of memory, use third page as output (two-way merge). 10/2=5 lists with 6 pages each
    c. Pass 3: Perform two-way merge again. 5/2=3 lists. First two lists have 12 pages, third list has 6.
    d. Pass 4: Repeat. 3/2=2 lists. One with 24, other with 6
    e. Pass 5: Repeat. 2/2 = 1 sorted list with 30 pages.
    f. 5*2*30=300 pages I/O cost
7. Schedules
    a. S1: No, No, No
    b. S2: No, Yes, Yes
    c. S3: No, Yes, Yes

8. Lock Tables

Data	Lock	Owner	Waiting
A	S	T1	

Data	Lock	Owner	Waiting
A	S	T1,T2	

Data	Lock	Owner	Waiting
A	S	T1,T2	

Data	Lock	Owner	Waiting
A	S	T1,T2	T1(X)

Data	Lock	Owner	Waiting
A	S	T1,T2	T1(X)

Data	Lock	Owner	Waiting
A	S	T1,T2	T1(X), T1(W(A))

Data	Lock	Owner	Waiting
A	S	T2	

Data	Lock	Owner	Waiting
A	X	T2	

Data	Lock	Owner	Waiting
A	X	T2	

Data	Lock	Owner	Waiting

9. MGL
    a. T1
        i. IS on db GRANTED
        ii. IS on f1 GRANTED
        iii. IS on p1 GRANTED
        iv. S on r3 GRANTED
    b. T2
        i. IX on db GRANTED
        ii. IX on f2 GRANTED
        iii. IX on p3 REJECTED (already has S)
        iv. X on r6 REJECTED (parent has no IX or SIX)

10. R-tree
    a. Overlap Search
        i. (R1, R2)
        ii. (R3, R4, R5), (R6, R7)
        iii. (R8, R9, R10)
    b. R1, R5

11. Data Mining
    a. 2/6 = .33333333333333333333333333333333…
    b. Confidence: 2/3; Support: 2/6 = 1/3
    c. 2 item sets
        i. {HotDogs}, {Coke}
        ii. {HotDogs}, {Chips}
        iii. {Coke}, {Chips}