Tutorial 2

Functional Dependencies

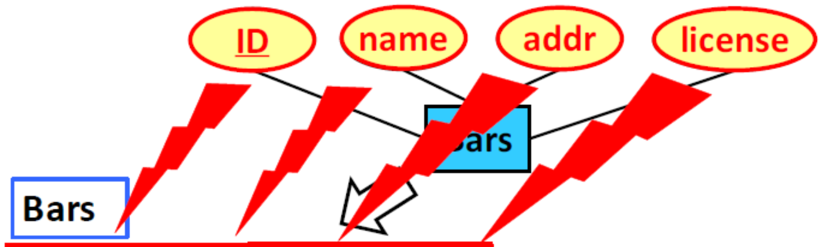
CZ2007
Introduction to Databases

Tutorial 2

ER Diagram → Relational Schema

- General rules:
  - Each entity set becomes a relation
  - Each many-to-many relationship becomes a relation
- Special treatment needed for:
  - Weak entity sets
  - Subclasses
  - Many-to-one and one-to-one relationships

Entity Set → Relation

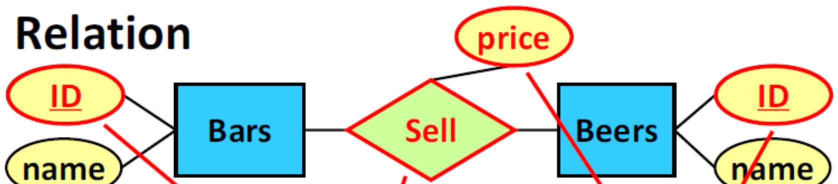


Bars

ID	name	addr	license
...	...	...	...

- Each entity set is converted into a relation that contains all its attributes
- Key of the relation = key of the entity set

Many-to-Many Relationship → Relation

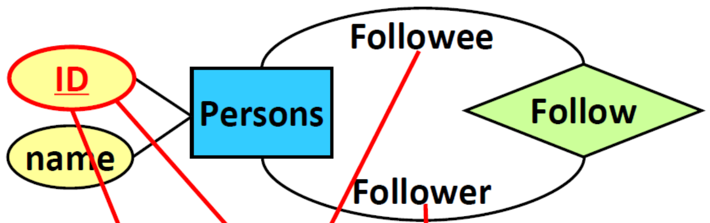


- Converted into a relation that contains
  - all keys of the participating entity sets, and
  - the attributes of the relationship (if any)
- Key of relation = Keys of the participating entity sets

Sell

Bars-ID	Beers-ID	price
...	...	...

Many-to-Many Relationship → Relation



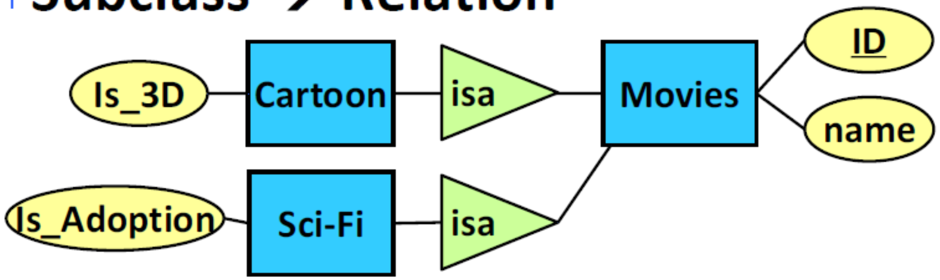
- If an entity is involved multiple times in a relationship
  - Its key will appear in the corresponding relation multiple times
  - The key is re-named according to the corresponding role

Follow

Followee-ID	Follower-ID
...	...

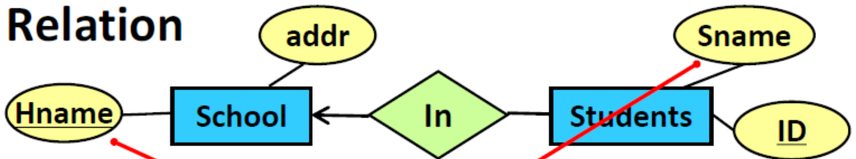
Tutorial 2

Subclass → Relation



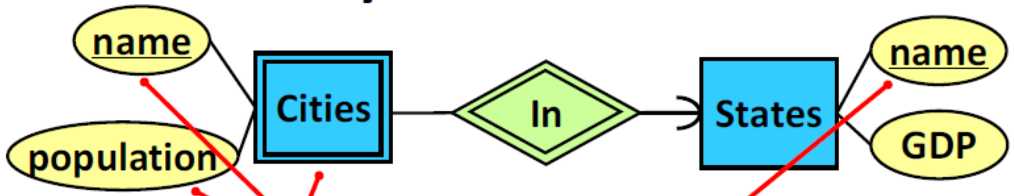
- One relation for each entity set
  - Movies(ID, name)
  - Cartoon(ID, Is_3D)
  - Sci-Fi(ID, Is_Adoption)

Many-to-One Relationship → Relation



- Translation:
  - School(Hname, addr)
  - Students(ID, Sname, Hname)
- Only need to put the key of the "one" side into the relation of the "many" side

Weak Entity Set → Relation

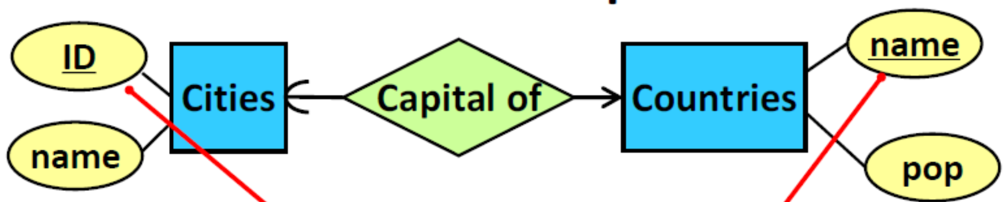


- Each weak entity set is converted to a relation that contains
  - all of its attributes, and
  - the key of the supporting entity set
  - attribute (if any) of supporting relationship
- The supporting relationship is ignored

Cities

state-name	city-name	population
…	…	…

One-to-One Relationship → Relation



- No need to create a relation for a one-to-one relationship
- Only need to put the key of one side into the relation of the other

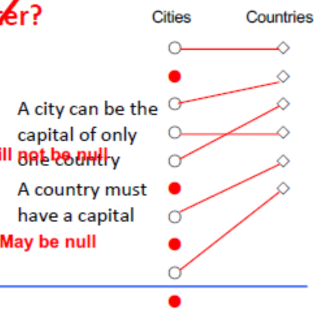
Sol 1 vs Sol 2: Which one is better?

- Solution 1
  - Cities(CityID, Cityname)
  - Countries(Countryname, pop, CityID)

Will not be null

- Solution 2
  - Cities(CityID, Cityname, Countryname)
  - Countries(Countryname, pop)

May be null



Tutorial 2

Functional Dependencies:

- In general, how do we know whether a combination of attributes is bad?

- We need to check the correlations among those attributes

Formal Definition of FD

- Attributes $A_1, A_2, \ldots, A_m$, $B_1, B_2, \ldots, B_n$
- $A_1A_2\ldots A_m \rightarrow B_1B_2\ldots B_n$
- Meaning: There do not exist two objects that
  - Have the same values on $A_1, A_2, \ldots, A_m$
  - but different values on $B_1, B_2, \ldots, B_n$

Reasoning with FDs

- Given $A \rightarrow B$, $BC \rightarrow D$
- Can you prove that $AC \rightarrow D$?
- Proof
  - Given $A \rightarrow B$, we have $AC \rightarrow BC$ (Augmentation)
  - Given $AC \rightarrow BC$ and $BC \rightarrow D$, we have $AC \rightarrow D$ (Transitivity)

Reasoning with FDs

- Armstrong's Axioms
  - Three axioms for FD reasoning
  - Easy to understand, but not easy to apply

- Axiom of Reflexivity
  - A set of attributes → A subset of the attributes

- Axiom of Augmentation
  - Given $A \rightarrow B$
  - We always have $AC \rightarrow BC$, for any C
- Axiom of Transitivity
  - Given $A \rightarrow B$ and $B \rightarrow C$
  - We always have $A \rightarrow C$

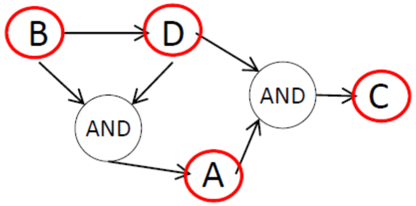
Tutorial 2

Reasoning with FDs

- Given A→B, BC→D
- Can you prove that AC → D?
- Proof
  - Given A→B, we have AC → BC (Augmentation)
  - Given AC→BC and BC → D, we have AC → D (Transitivity)

Steps of the Intuitive Solution

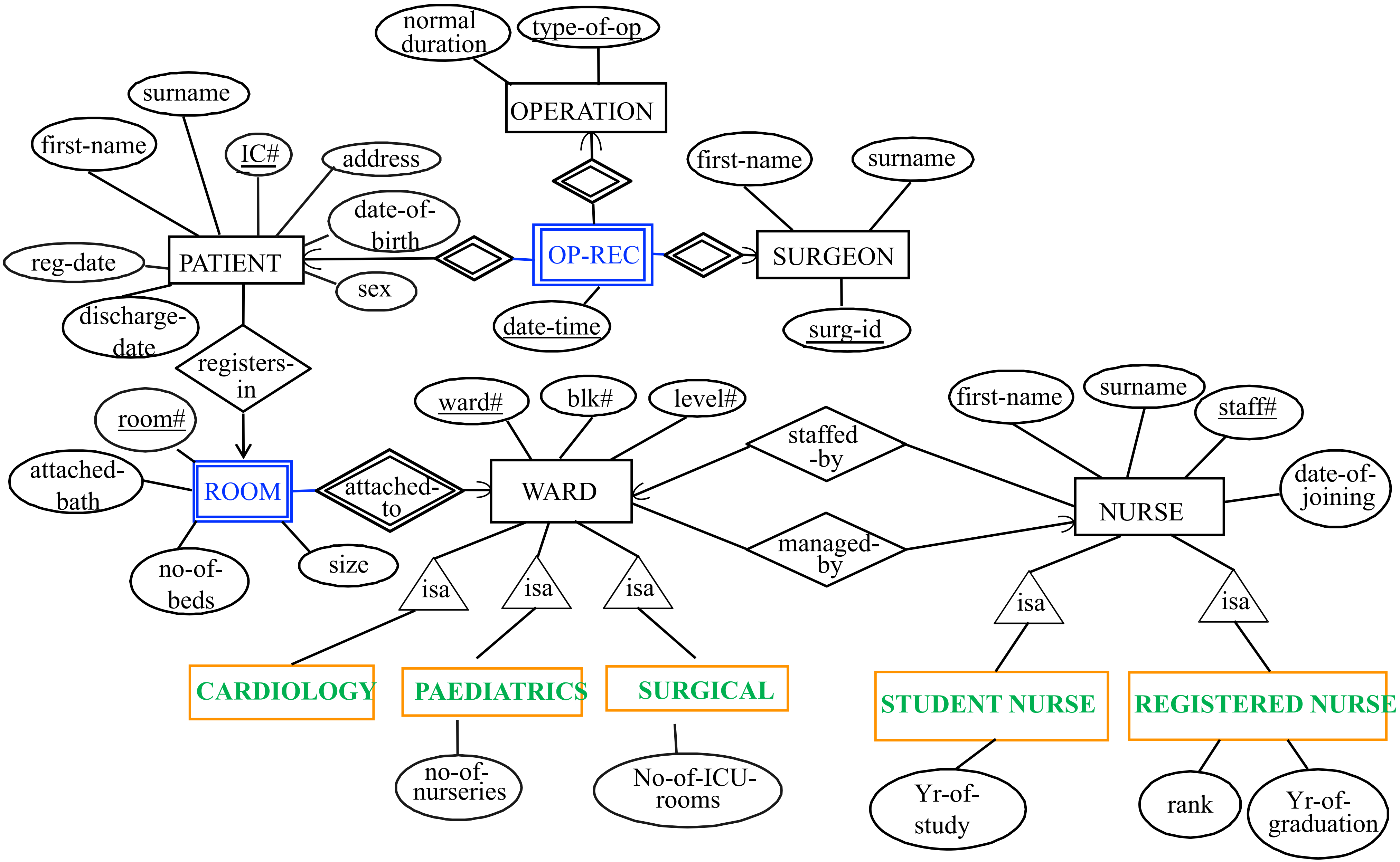
- Four attributes: A, B, C, D
- Given: B→D, DB→A, AD→C
- Can you prove B→C?



- First, activate B
  - Activated set = { B }
- Second, activate whatever B can activate
  - Activated set = { B, D }, since B→D
- Third, use all activated elements to activate more
  - Activated set = { B, D, A }, since DB→A
- Repeat the third step, until no more activation is possible
  - Activated set = { B, D, A, C }, since AD→C; done

Question 1

1. Translate the ER Diagram of Q1 in Tutorial 1 into a set of relations.



Question 1: Relational Tables

- Surgeon (<u>surg-id</u>, first-name, surname)
- Operation (<u>type-of-op</u>, normal-duration)
- Patient (<u>IC#</u>, first-name, surname, address, date-of-birth, sex, reg-date, discharge-date, room#, ward#) – m2o
- Ward (<u>ward#</u>, blk#, level#, no-of-nurseries, no-of-ICU-rooms, manager) – m2o
- Nurse (<u>staff#</u>, first-name, surname, date-of-joining, wardStaff#) – m2o
- Op-Rec (<u>IC#</u>, <u>surg-id</u>, <u>type-of-op</u>, <u>date-time</u>) – w.e./m2o
- Room (<u>room#</u>, <u>ward#</u>, attached-bath, no-of-beds) – w.e./m2o

Question 1: Relational Tables

- Subclass relationships:
- Student_Nurse(<u>staff#</u>, Yr-of-study)
- Registered_Nurse(<u>staff#</u>, rank, yr-of-graduation)
- Surgical_Ward(<u>ward#</u>, No_of_ICU_rooms)
- Pediatrics_Ward(<u>ward#</u>, No_of_nurseries)
- Cardiology_Ward(<u>ward#</u>, no_of_heart_equipment)

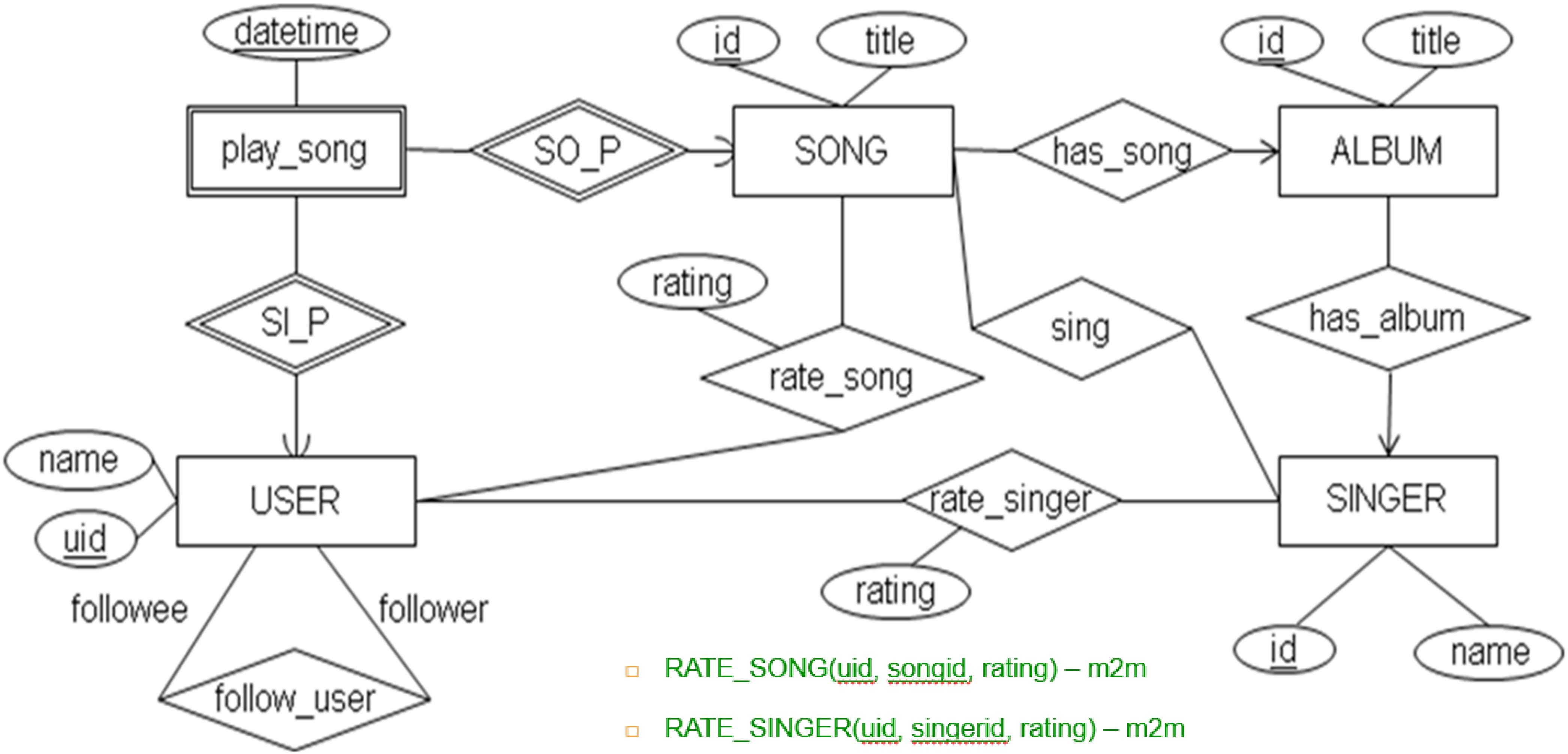
Question 2

2. Consider the following relational schema:

- **USER**(uid, name)
- **SINGER**(id, name)
- **ALBUM**(id, title, singerid) – m2o
- **SONG**(id, title, albumid) – m2o
- SING(singerid, songid) – **m2m**
- FOLLOW_USER(followeruid, followeeuid) – **m2m**
- RATE_SONG(uid, songid, rating) – **m2m**
- RATE_SINGER(uid, singerid, rating) – **m2m**
- PLAY_SONG(uid, songid, datetime) – weak entity m2o

Construct an ER diagram that leads to the above schema.

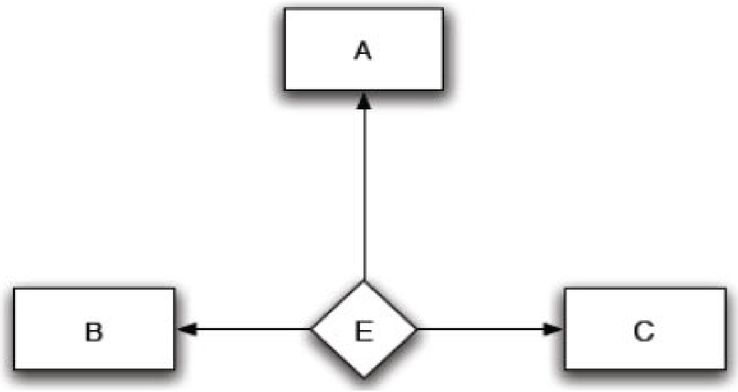
Question 2



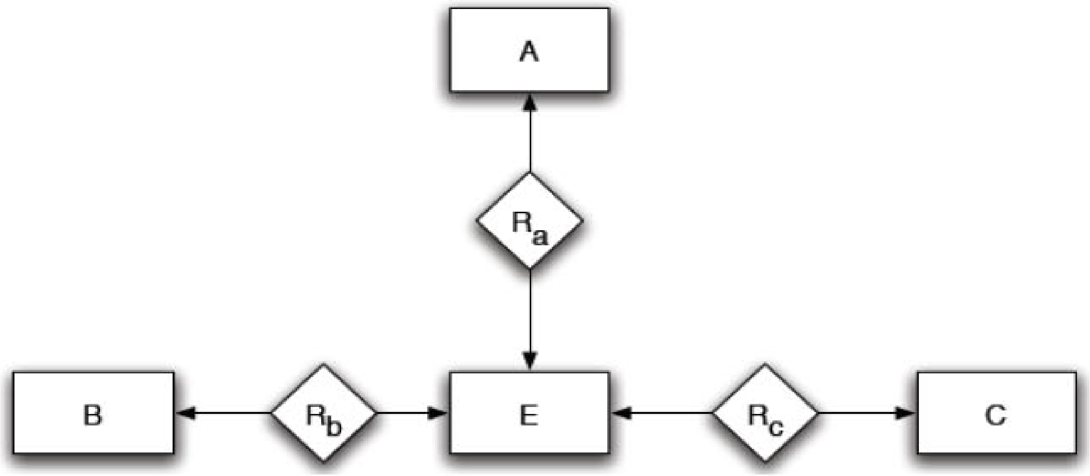
- RATE_SONG(uid, songid, rating) – m2m
- RATE_SINGER(uid, singerid, rating) – m2m
- PLAY_SONG(uid, songid, datetime) – weak entity m2o
- SING(singerid, songid) – m2m
- FOLLOW_USER(followeruid, followeeuid) – m2m
- SINGER(id, name)
- USER(uid, name)
- SONG(id, title, albumid) – m2o
- ALBUM(id, title, singerid) – m2o

Question 3

3. The figure shows an attempt to represent a ternary relationship between three entities using 3 binary relationships (and one made-up entity). Show through an example why the 3 binary relationship representation is more general than the one ternary relationship representation.



A ternary relationship



An attempt to represent it using 3 binary relationships

Question 3

Why is a **3 binary relationship relationship representation** more general than **one ternary relationship representation**? (https://stackoverflow.com/questions/39254141/ternary-relationship-or-3-binary-relationship)

Because with 3 binary relationships each involved entity is related **separately** with each one of the others two.

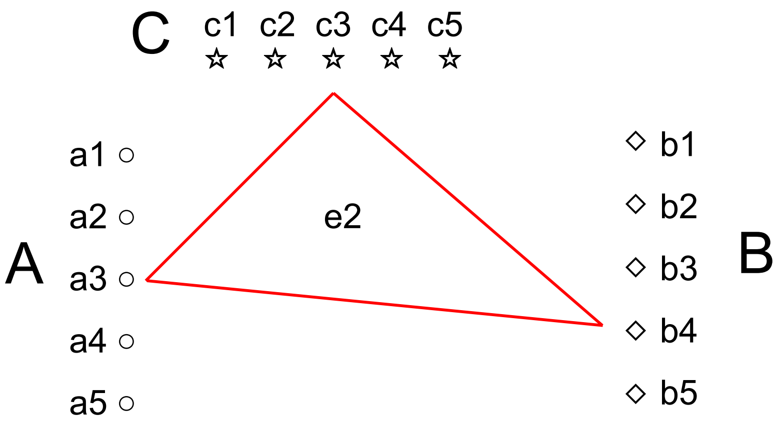
Assumed that ternary relationship in essential only when you have a many-to-many-to-many relationship (otherwise you can rewrite it adding a relational entity linked with binary relationships to the others three), let's take an example:

- Suppose that you have the entities A, B and C, and three relationships that link the entities "like a triangle". Now, suppose that set of tuples a3 is related to the sets b4 and c3.
- Using 3 binary relationships, it's not requested that b4 and c3 are related. for example, b4 can be related to c2, that can partially overlap c3 or be completely disjoined.
- With a ternary relationship, instead, b4 must be related to c3.
- So you can see that with 3 binary relationships you can have much more combinations.

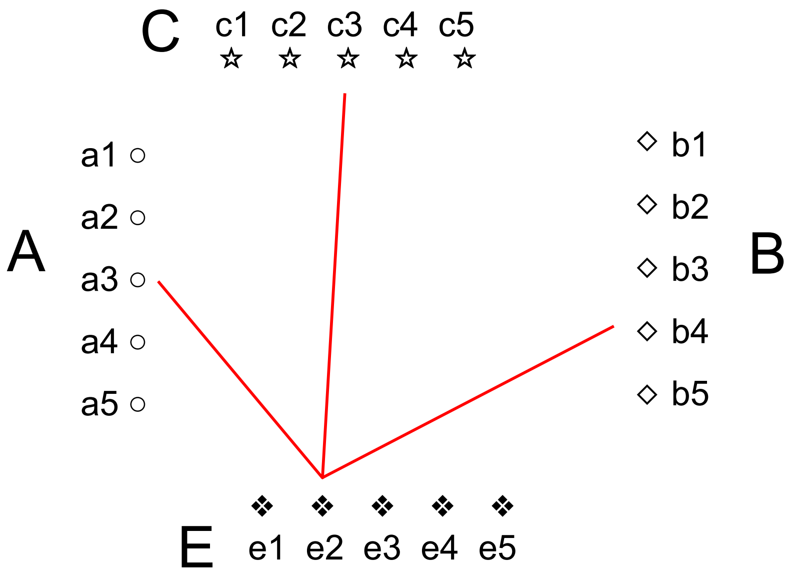
Question 3

Ternary relationship

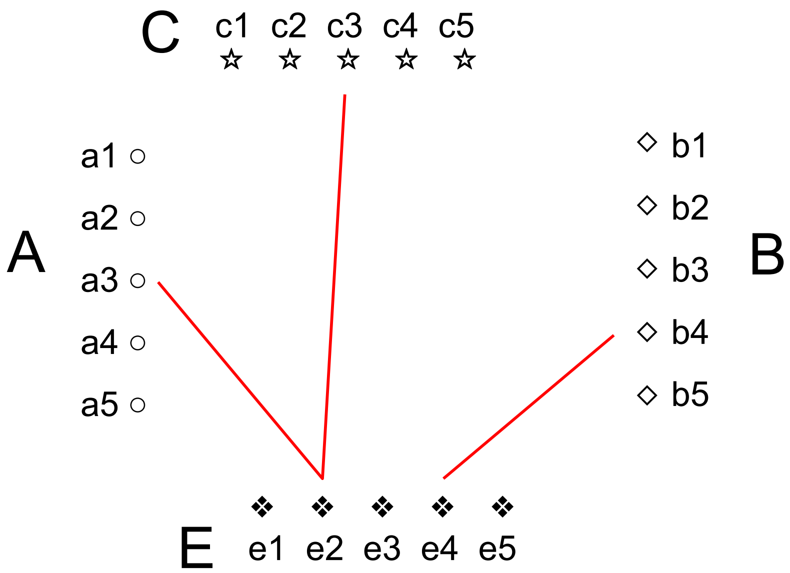
Three 2-binary relationship



=

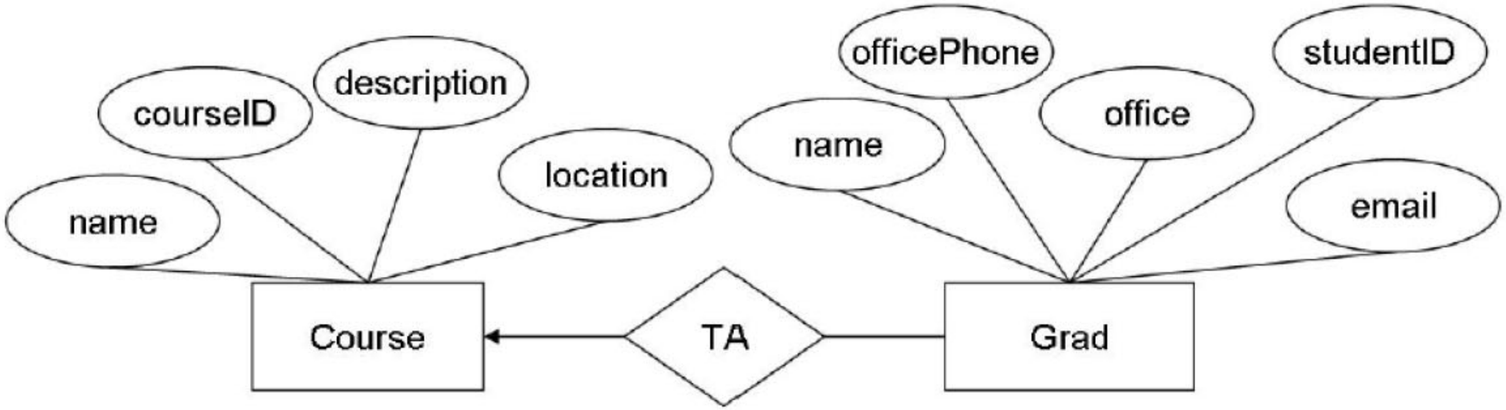


?



Question 4

4. Consider the following ER diagram thath describes graduate students (Grad) and courses (Course) they serve as Teaching Assistants (TA).



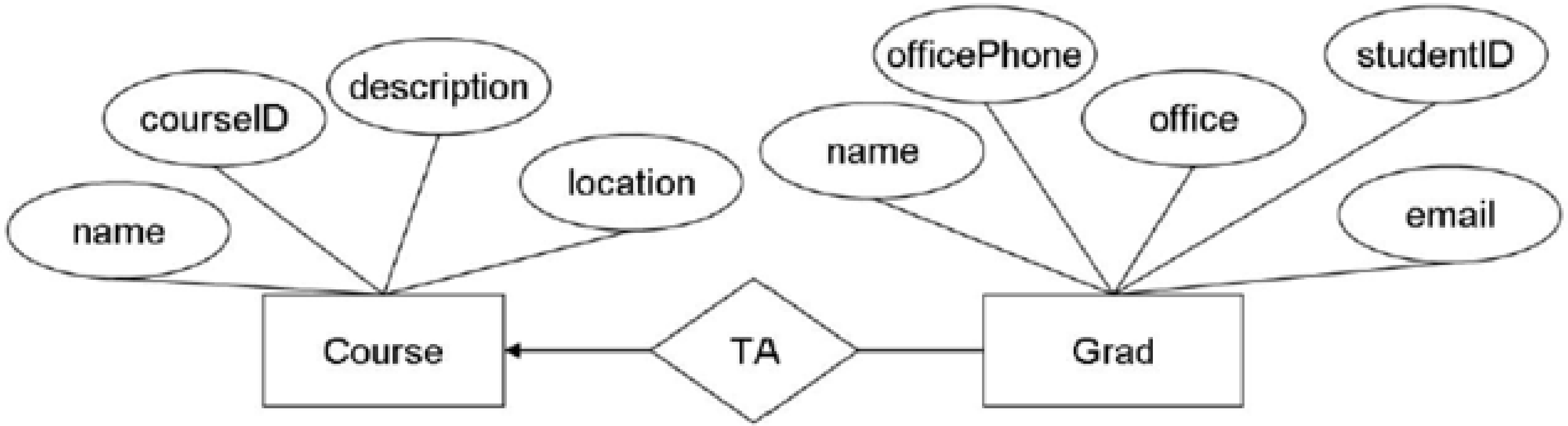
(a) For each of the following statements, write a functional dependency (FD) that best captures the statement.

- The studentID of each graduate student uniquely identifies the student.
- No two offices have the same phone number (officePhone).
- No two courses have the same courseID.
- If two courses have the same course name, their course descriptions are the same.

(b) From the ER diagram and the set of FDs you listed above, can you derive new FDs? If no, explain why not. If yes, derive two non-trivial FDs.

Question 4(a)

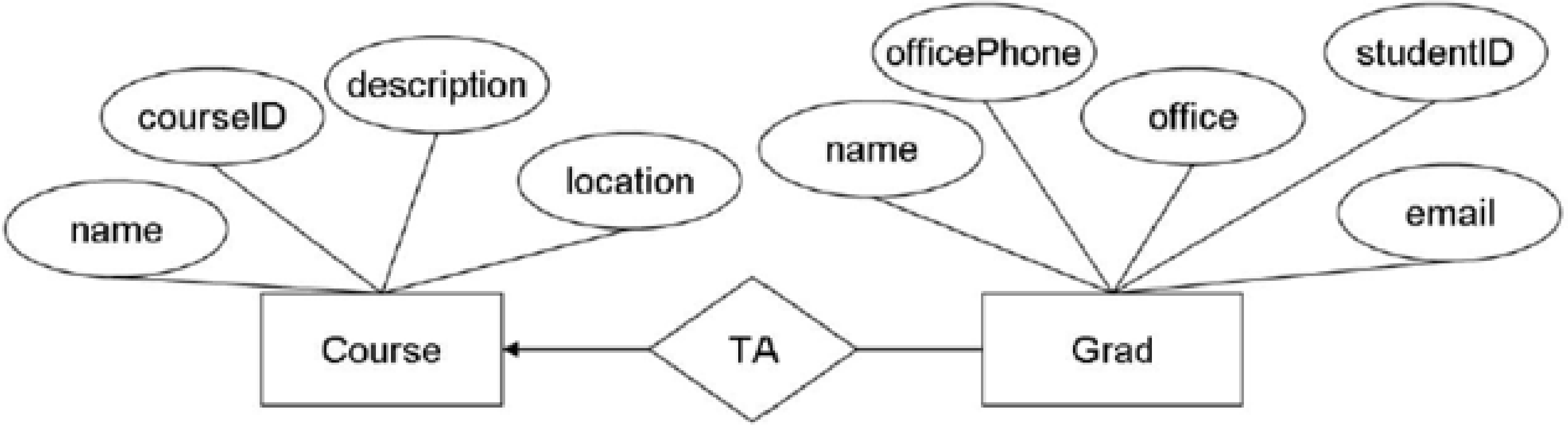
(a) For each of the following statements, write a functional dependency (FD) that best captures the statement.



- The studentID of each graduate student uniquely identifies the student.
  - F1: studentID → officePhone, office, email, name
- No two offices have the same phone number (officePhone).
  - F2: officePhone → Office

Question 4(a)

(a) For each of the following statements, write a functional dependency (FD) that best captures the statement.



- No two courses have the same courseID.

- F3: courseID → name, description, location

- If two courses have the same course name, their course descriptions are the same.

- F4: name → description

Question 4(b)

(b) From the ER diagram and the set of FDs you listed above, can you derive new FDs? If no, explain why not. If yes, derive two non-trivial FDs.

- F5: studentID $\rightarrow$ courseID.

  This is derived from the TA relationship, F1, and F3.

- F6: studentID $\rightarrow$ name, description, location.

  This is derived from F3 and F5.