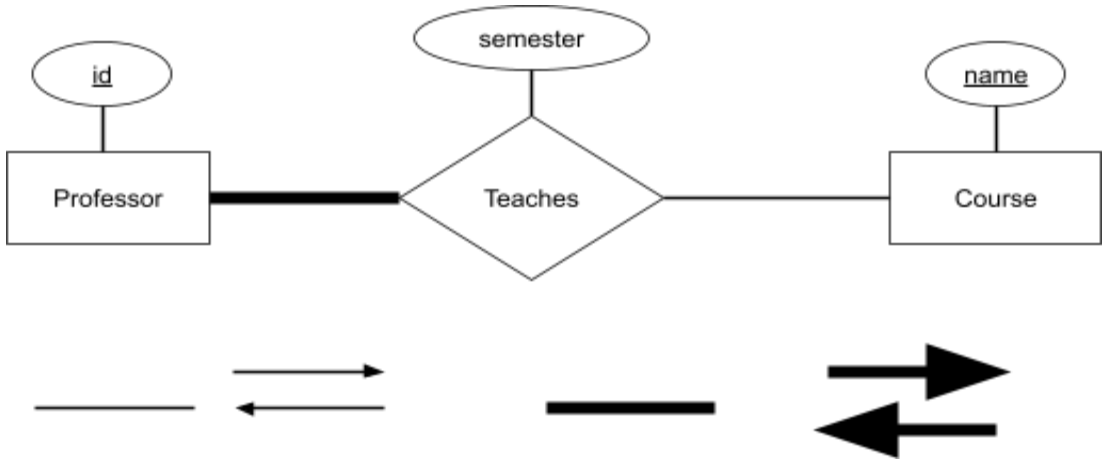
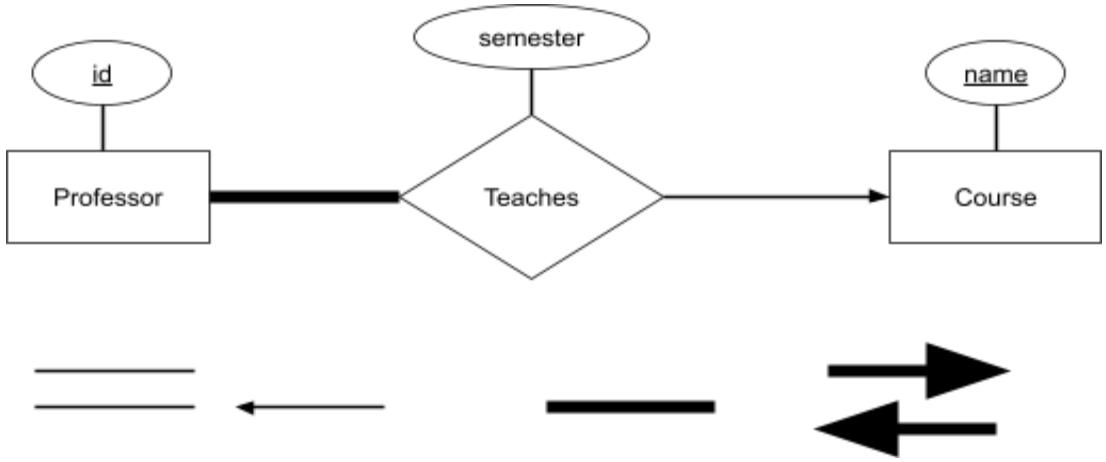
Problem Set 1: Part I

Problem 1: ER diagram basics

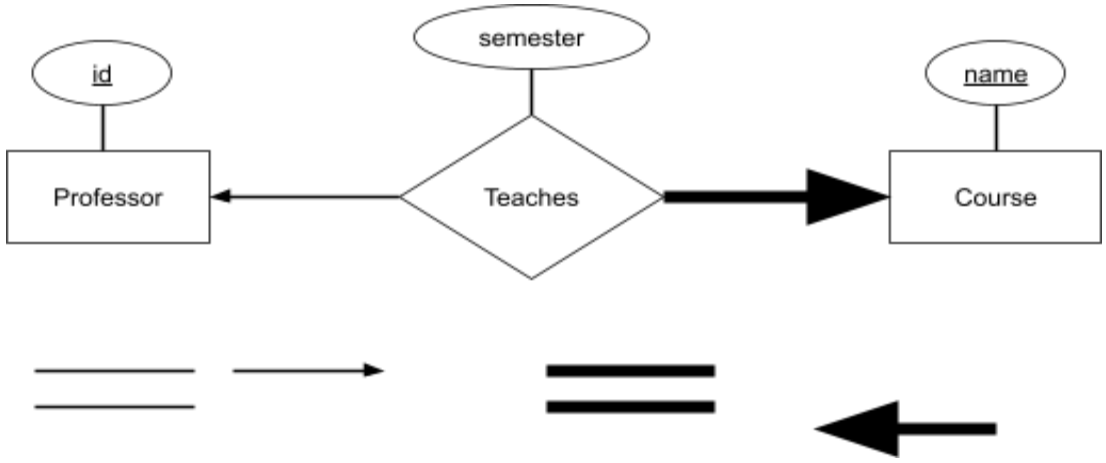
1.



2.



3.



Problem 1 (cont)

4.

Yes

According to the question, we know that it is many-to-many relation from course to professor

For a many-to-many relationship set, we take the union of the primary keys of the connected entity sets

The primary key of Professor is id, and the primary key of Course is name, so

Teaches(professor, course, semester)

where *professor* is a foreign key referring to *Professor(id)*, *course* is a foreign key referring to *Course(name)*, and the primary-key attributes of *Teaches* are underlined

is an acceptable schema for a relation used to capture the Teaches relationship set

Problem 2: Database design

1. *BelongsTo* is a many-to-one relationship from *Artist* to *Label*

2. Each artist sings at least one songs

At least one song appear on each album

Each label produces at least one Album

At least one artist belong to at most one label (Each label has at least one artist and each artist belongs to at most one label)

3.

Relations that store info. about a type of entity:

Artist(name, dob, _id_)

Song(name, duration, _id_)

Album(_id_, name)

Label(_name_, address)

Relations that capture relationships between entities:

Sings(_artist_, _song_)

ApperasOn(_song_, _album_)

Produces(_label_, _album_)

BelongsTo(_artist_, label)

where

artist is a foreign key referring to *Artist(id)*,

each value of the artist attribute must match a value of the id attribute from the Artist relation;

song is a foreign key referring to *Song(id)*,

each value of the song attribute must match a value of the id attribute from the song relation;

album is a foreign key referring to *Album(id)*,

each value of the album attribute must match a value of the id attribute from the album relation;

label is a foreign key referring to *Label(name)*,

each value of the label attribute must match a value of the name attribute from the label relation

From Q1, we know that *BelongsTo* is a many-to-one relationship from *Artist* to *Label*, so we can eliminate the relation for the relationship set (*BelongsTo*) and capture the relationship set in the relation used for the entity set on the *many* side of the relationship (*Artist*):

Relations that store info. about a type of entity:

Artist(name, dob, _id_, label)
Song(name, duration, _id_)
Album(_id_, name)
Relations that capture relationships between entities:
Sings(_artist_, _song_)
ApperasOn(_song_, _album_)
Produces(_label_, _album_)
where
artist is a foreign key referring to *Artist(id)*,
Each value of the artist attribute must match a value of the id attribute from the Artist relation and the name attribute from the Label relation;
song is a foreign key referring to *Song(id)*,
each value of the song attribute must match a value of the id attribute from the Song relation;
album is a foreign key referring to *Album(id)*,
each value of the album attribute must match a value of the id attribute from the Album relation;
label is a foreign key referring to *Label(name)*,
each value of the label attribute must match a value of the name attribute from the label relation

Problem 3: Combining relations
Use the Insert->Table menu option to insert an appropriate table for each answer.

1.

R_a	R_b	c	S_a	S_b
1	2	3	2	3
1	2	3	3	4
1	2	3	7	6
3	4	3	2	3
3	4	3	3	4
3	4	3	7	6
7	6	5	2	3
7	6	5	3	4
7	6	5	7	6

2.

a	b	c
3	4	3

7	6	5

3.

a	b	c
3	4	3
7	6	5
1	2	3

4.

a	b	c
3	4	3
7	6	5
2	3	null

5.

a	b	c
3	4	3
7	6	5
1	2	3
2	3	null