COMP3311 Week 3

Here's a cute image I generated w/ Midjourney AI →

The prompt was:

"Kitten doing ballet over the moon"

By Manhua Lu

Announcements

- Quiz 2 due Friday 1 March @11:59pm AEST
- Week 2 Tutorial answers + Quiz 1 results available now on WebCMS
- You should have PostgreSQL set up by now
    - Please let me know if you do not have it set up and need help

Learning Objectives

01 → ER to Relational Mapping (Q2, Q3)

02 → SQL Data Definition Language (Q7, Q8, Q9)

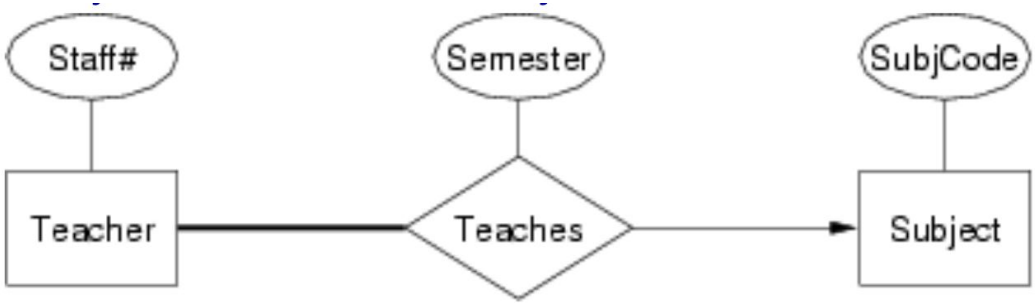
03 → ER to SQL Mapping (Q13, Q17, Q20)

01

ER to Relational Mapping

01 → ER to Relational Mapping: ER Recap

- **Entities**: Objects (rectangles)
- **Attributes**: Properties (oval)
  - Primary key is underlined
- **Relationships**:
  - Cardinality (one to many, one to one, many to many)
  - Totality (participation)

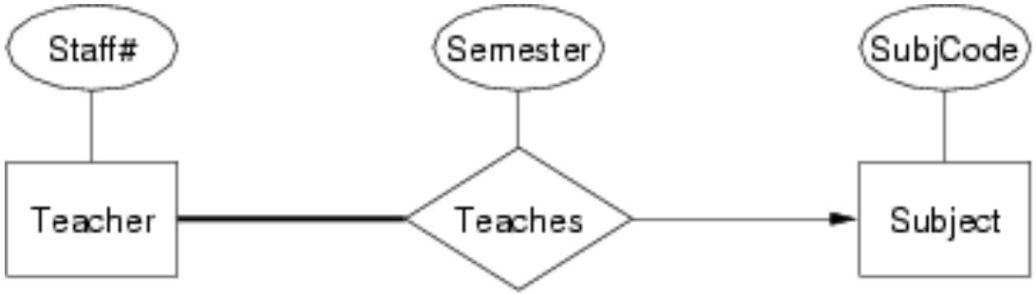


01 → ER to Relational Mapping: ER Recap

- **Entities**: Objects (rectangles)
- **Attributes**: Properties (oval)
  - Primary key is underlined
- **Relationships**:
  - Cardinality (one to many, one to one, many to many)
  - Totality (participation)

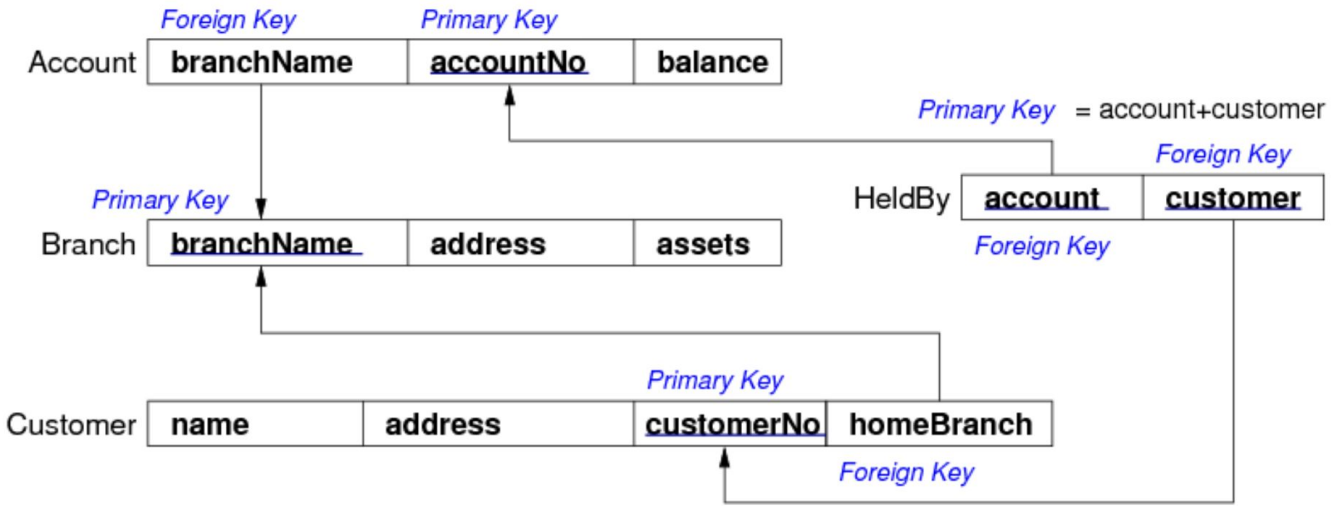
One to many

"Teacher must teach ONE subject. A subject can be taught by 0 or more teachers"



01 ER to Relational Mapping: Relational Recap

- Model data as a set of relations with attributes
- General process is to **create a relational model from ER diagram**



01 ER to Relational Mapping

- Entities become tables
- Relationships:

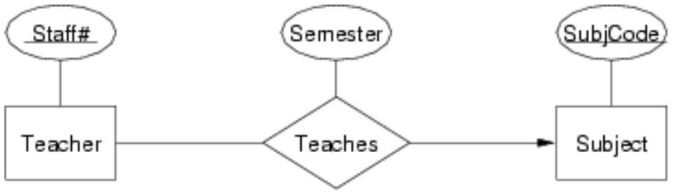
Many to many	Relations become a separate table with foreign keys (fk)
One to one	Relations collapse into either entity, depends on participation. Usually put in the entity with total participation
One to many	Relationship attribute is put in the ‘one’ side Foreign key of the ‘many’ side is also put in the ‘one’ entity

- Different types of mapping:
  - ER Style
  - Object-Oriented Style
  - Single table with nulls

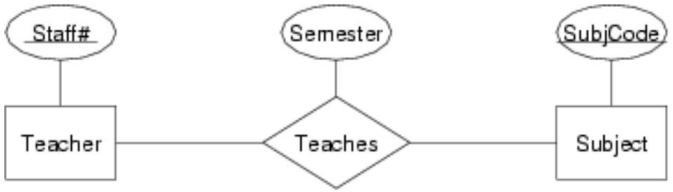
01 ER to Relational Mapping Question

2. Convert each of the following ER design fragments into a relational data model expressed as a box-and-arrow diagram:

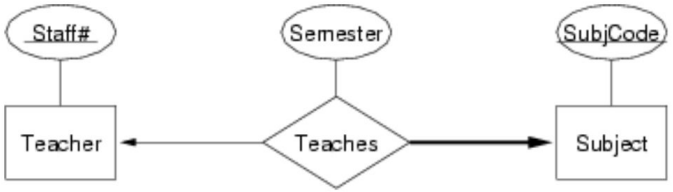
a.



b.



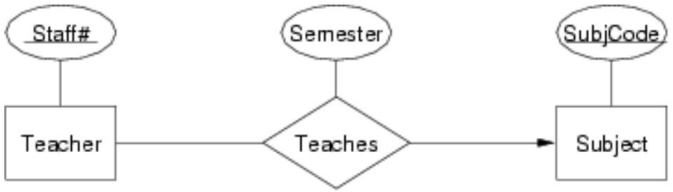
c.



01 ER to Relational Mapping Question

2. Convert each of the following ER design fragments into a relational data model expressed as a box-and-arrow diagram:

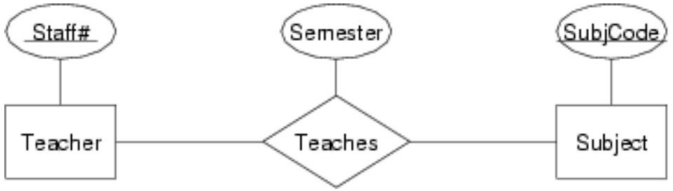
a.



One to many

"Teacher can teach 0 or 1 subject. A subject can be taught by 0 or more teachers"

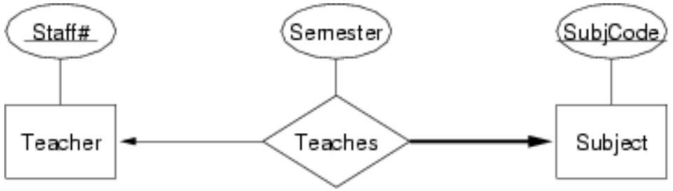
b.



Many to many

"Teacher can teach 0 or more subject. A subject can be taught by 0 or more teachers"

c.

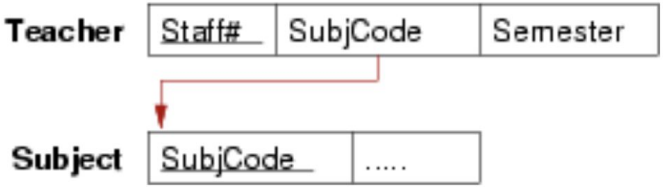


One to one

"Teacher can teach 0 or 1 subjects. A subject must be taught by 1 teacher"

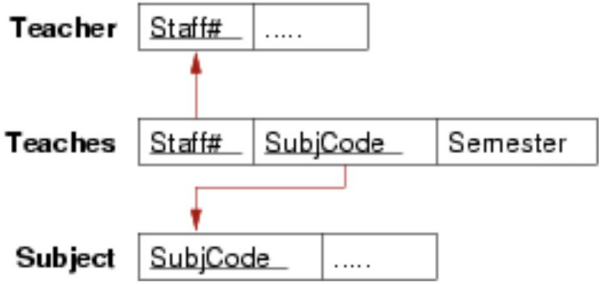
01 → ER to Relational Mapping Answer

a) One to many relationship. ‘Semester’ attribute is put in the ‘One’ side. Fk ‘SubjCode’ in ‘One’ side as well.



b) Many to many relationship, so we have a separate table for ‘Teaches’

The primary key for the Teaches relationship table is Staff# + SubjCode

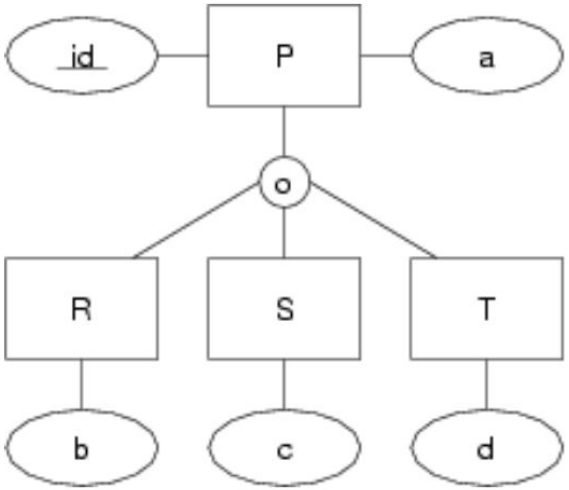


c) One to one relationship. Putting in either side is usually fine, but because of the **totality** best to put in the side with the total participation.

i) This ensures that ‘Staff#’ will never be NULL

01 → ER to Relational Mapping Question

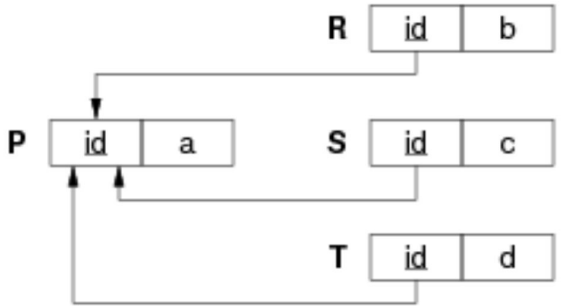
3. In the mapping from the ER model to the relational model, there are three different ways to map class hierarchies (ER, OO, single-table). Show each of them by giving the mapping for the following class hierarchy:



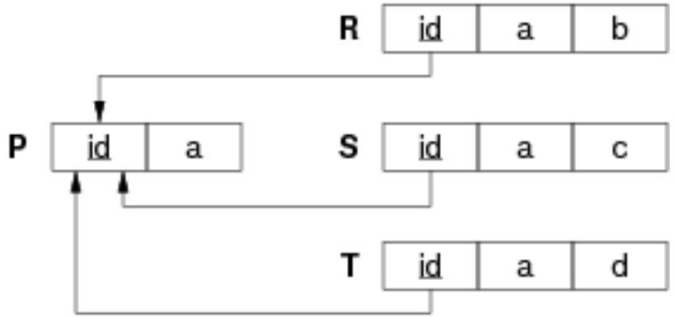
Use box-and-arrow diagrams for the relational models.

01 → ER to Relational Mapping Answer

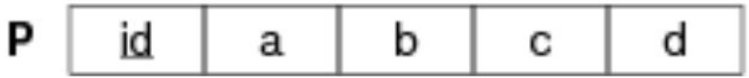
a) ER-Style Mapping



b) Object-oriented-style Mapping (attributes from parent class are put in the subclass' table)



c) Single-table-Style Mapping (attribute is null if unused)



02

SQL Data Definition Language

02 → SQL Data Definition Language

- In SQL we can model relation schemas as tables

```
create table TableName (
    attribute1 domain1 constraints1,
    attribute2 domain2 constraints2,
    ...
    table-level constraints,...
);
```

02 → SQL Data Definition Language Examples

```
create table Students(
    zid         serial,
    familyName  varchar(40) not NULL,
    givenName   varchar(30) not NULL,
    dob         date not NULL,
    gender      char(1) check (gender in ('M', 'F', 'X')),
    degree      integer,
    primary key (zid),
    foreign key (degree) references Degrees(dID)
);
```

```
create domain GenderType as
    char(1) check (value in ('M', 'F', 'X'));

create table Students(
    zid         serial primary key,
    familyName  varchar(40) not NULL,
    givenName   varchar(30) not NULL,
    dob         date not NULL,
    gender      GenderType,
    degree      integer references Degrees(dID)
);
```

02 SQL Data Definition Language Question

7. What is the difference between the following two ways to define a primary key?

```
create table R (
   a integer primary key,
   b integer,
   ...
);
```

```
create table R (
   a integer,
   b integer,
   ...
   primary key (a)
);
```

02 → SQL Data Definition Language Question

7. What is the difference between the following two ways to define a primary key?

```
create table R (
   a integer primary key,
   b integer,
   ...
);
```

```
create table R (
   a integer,
   b integer,
   ...
   primary key (a)
);
```

No difference for a single attribute primary key (as above)

BUT you cannot define a multi-attribute primary key inline e.g. →

```
-- this does not work

create table R (
   a integer primary key,
   b integer primary key,
   ...
);
```

```
-- define it like this

create table R (
   a integer,
   b integer,
   ...
   primary key (a,b)
);
```

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

a. people's names

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

a. people's names

- Depends on how the names are going to be used.
  - Store them as a single string in the format "familyName, givenNames"

```
name        varchar(40)
```

OR

  - Store them as the single strings

```
familyName  varchar(30) not NULL,
givenName   varchar(30) not NULL
```

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

b. addresses

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

b. addresses

- Addresses could be broken down into:

```
street  varchar(30),
town    varchar(30),
state   varchar(30),
country varchar(30)
```

OR

- Single string

```
address  varchar(80)
```

OR

- To check country is valid, country can be a separate table

```
street  varchar(30),
town    varchar(30),
state   varchar(30),
country integer references Country(id)
```

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

c. ages

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

c. ages

- Probably better to use date-of-birth rather than age because age changes over time
    - With date-of-birth we can compute age
- Although if someone insists on having an age attribute then:

```
age  integer check (age > 0 and age < 150)
```

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

d. dollar values

02 → SQL Data Definition Language Question

8. Discuss suitable SQL representations for each of the following attributes, including additional domain constraints where relevant:

d. dollar values

- For monetary values, we need an arbitrary number of total digits, with two digits after the decimal float. This could be done as: `dollarValue numeric(20,2)`

  OR

- `dollarValue float`

  OR

- Some database systems (e.g. PostgreSQL) have special types:

  `dollarValue money`

02 → SQL Data Definition Language Question

9. In many real PostgreSQL schemas, you will see definitions like

```
create table R (
   id    serial,
   name  text,
   d_o_b date,
   ...
   primary key (id)
);
```

a. What is the effect of the serial declaration?

02 → SQL Data Definition Language Question

9. In many real PostgreSQL schemas, you will see definitions like

```
create table R (
   id    serial,
   name  text,
   d_o_b date,
   ...
   primary key (id)
);
```

a. What is the effect of the serial declaration?

- It creates an integer attribute
- Every time you insert a new tuple and don't give a value for the id attribute, the sequence supplies a new one and increments itself. (e.g. on next slide)
  - The default serial starting number is 1.
  - A table can have no more than one SERIAL column

02 → SQL Data Definition Language Question

9. In many real PostgreSQL schemas, you will see definitions like

```
create table R (
   id    serial,
   name  text,
   d_o_b date,
   ...
   primary key (id)
);
```

b. How would you make use of it when inserting tuples?

02 → SQL Data Definition Language Question

9. In many real PostgreSQL schemas, you will see definitions like

```
create table R (
   id    serial,
   name  text,
   d_o_b date,
   ...
   primary key (id)
);
```

b. How would you make use of it when inserting tuples?

- You need to use the returning clause to capture the generated value, e.g. `insert into R(name,d_o_b) values ('John','1972-02-28') returning id;`
  - Note that we don't give the tuple the id, so the sequence supplies a new one and increments itself.

02 → SQL Data Definition Language Question

9. In many real PostgreSQL schemas, you will see definitions like

```
create table R (
   id    serial,
   name  text,
   d_o_b date,
   ...
   primary key (id)
);
```

c. How would you reference R.id as a foreign key?

02 → SQL Data Definition Language Question

9. In many real PostgreSQL schemas, you will see definitions like

```
create table R (
   id    serial,
   name  text,
   d_o_b date,
   ...
   primary key (id)
);
```

c. How would you reference R.id as a foreign key?

- Since the serial attribute contains an integer value, you would reference it as e.g.

```
RId          integer references R(id)
```

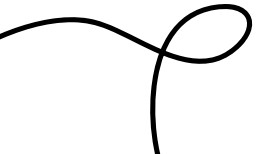
  - Defining the RId attribute as serial actually works, but generates a useless integer as a side-effect.

03

ER to SQL Mapping

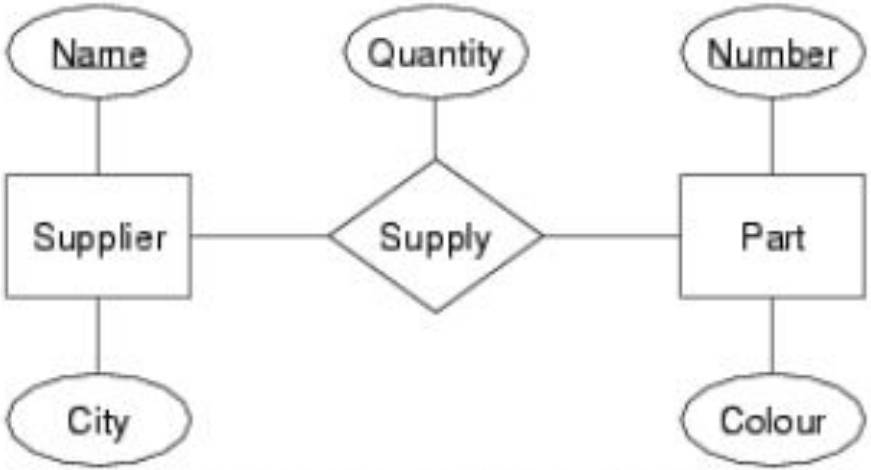
03 → ER to SQL Mapping

- General Steps:
  1. Convert ER to a relational diagram
  2. Convert relational diagram to SQL
  3. Have both diagrams to **ensure the SQL obeys all the constraints set out** in the ER diagram



03 → ER to SQL Mapping Question

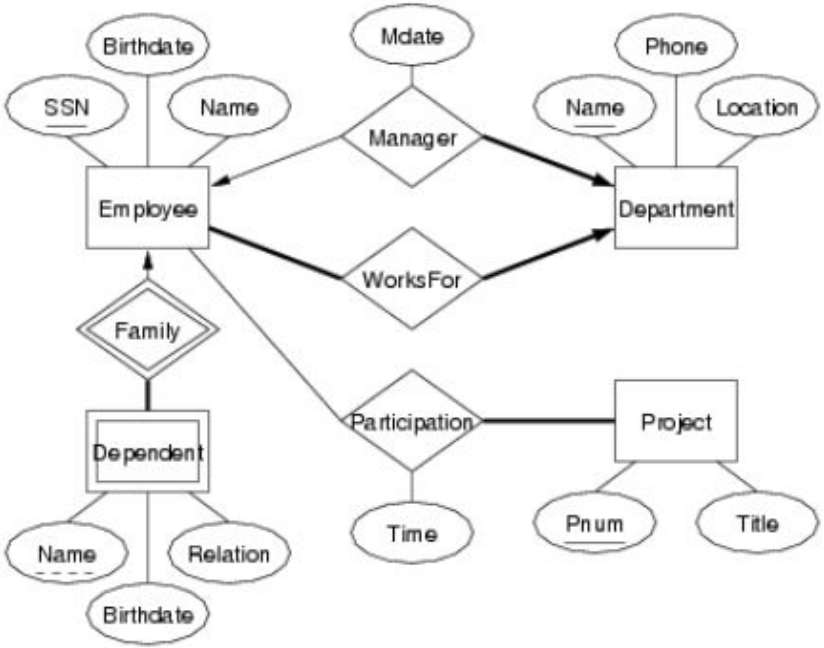
13. Convert the following ER design into an SQL schema:



Which elements of the ER design do not appear in the relational version?

03 → ER to SQL Mapping Question

17. Convert the following ER design to relational form:

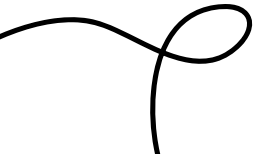


Which elements of the ER design do not appear in the relational version?

03 → ER to SQL Mapping Answer

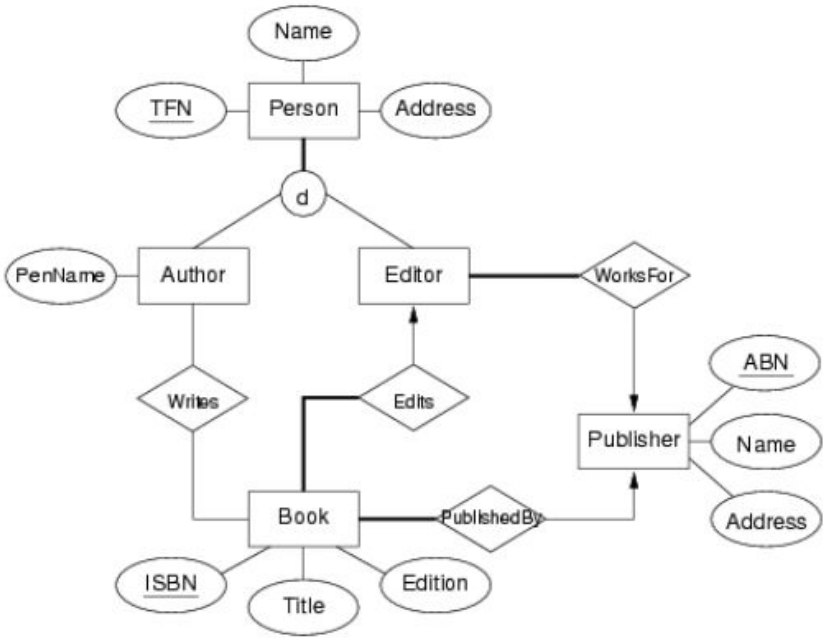
Elements of ER design that **do not** appear in the relational version:

- The total participation (but these can appear in the SQL schema as not NULL):
  - Department needing ONE manager
  - Employee needing to work for ONE department



03 → ER to SQL Mapping Question

20. Convert this ER design for the book publishing scenario into an SQL schema:



Give two versions, one using the ER-style mapping of subclasses, and the other using single-table-with-nulls mapping of subclasses.

03 → ER to SQL Mapping Answer

SQL Schema: using ER-style mapping of subclasses

```
create domain TaxFileNum as char(11)
                check (value ~ '^[0-9]{3}-[0-9]{3}-[0-9]{3}$');
create domain ISBNumber as char(15)
                check (value ~ '^[A-Z][0-9]{3}-[0-9]{4}-[0-9]{5}$');
create domain ABNumber as integer check (value > 100000);

create table Publisher (
        abn         ABNumber,
        name        varchar(60),
        address     varchar(100),
        primary key (abn)
);

create table Person (
        tfn         TaxFileNum,
        name        varchar(50),
        address     varchar(100),
        primary key (tfn)
);

create table Author (
        person      TaxFileNum,
        penname     varchar(50),
        primary key (person),
        foreign key (person) references Person(tfn)
);
```

```
create table Editor (
        person      TaxFileNum,
        publisher   ABNumber not null,
        primary key (person),
        foreign key (person) references Person(tfn),
        foreign key (publisher) references Publisher(abn)
);

create table Book (
        isbn        ISBNumber,
        title       varchar(100),
        edition     integer check (edition > 0),
        editor      TaxFileNum not null,
        publisher   ABNumber not null,
        primary key (isbn),
        foreign key (editor) references Editor(person),
        foreign key (publisher) references Publisher(abn)
);

create table Writes (
        author      TaxFileNum,
        book        ISBNumber,
        primary key (author,book),
        foreign key (author) references Author(person),
        foreign key (book) references Book(isbn)
);
```

03 → ER to SQL Mapping Answer

SQL Schema: using single-table-style mapping of subclasses

```
-- Uses single-table-style mapping for subclasses of Person

create domain TaxFileNum as char(11)
                check (value ~ '^[0-9]{3}-[0-9]{3}-[0-9]{3}$');
create domain ISBNumber as char(15)
                check (value ~ '^[A-Z][0-9]{3}-[0-9]{4}-[0-9]{5}$');
create domain ABNumber as integer check (value > 100000);

create table Publisher (
        abn         ABNumber,
        name        varchar(60),
        address     varchar(100),
        primary key (abn)
);

create table Person (
        tfn         TaxFileNum,
        name        varchar(50),
        address     varchar(100),
        kind        varchar(10) check (kind in ('author','editor')),
        -- attributes for Authors
        penname     varchar(50),
        -- attributes for Editors
        publisher   ABNumber not null,
        primary key (tfn),
        foreign key (publisher) references Publisher(abn),
        constraint  NoPenNameIfEditor check
                        ((kind = 'author' and publisher is null) or
                         (kind = 'editor' and penname is null))
);
-- Problem with the above:
-- * publisher attribute defined to be not null
-- * if author type, publisher is required to be null
-- * to resolve this, we have to lose one of the constraints
--   - either lose total participation of Editor with Publisher
--   - or lose check on null poublisher for authors
```

```
create table Book (
        isbn        ISBNumber,
        title       varchar(100),
        edition     integer check (edition > 0),
        editor      TaxFileNum not null,
        publisher   ABNumber not null,
        primary key (isbn),
        foreign key (editor) references Person(tfn),
        foreign key (publisher) references Publisher(abn)
);

create table Writes (
        author      TaxFileNum,
        book        ISBNumber,
        primary key (author,book),
        foreign key (author) references Person(tfn),
        foreign key (book) references Book(isbn)
);
```