Conceptual Database Design

Textbook: chapters 3 & 4

22T1; Week1; Feb17th

Scenario

A: “Welcome to the job, can you build us a database to organize all the information on a publisher and present it at the meeting?”

B: “okay, what’ll be in the database?”

A: “Well… authors, books, editors, printers etc. An author can write zero or more books and a book is written by one or more authors. Where a book is uniquely identified by its book-id. For each book, we also record its title, price, and availability. An editor is uniquely identified by his/her reader-id and we also record his/her name, phone number, address. The address is composed of street and suburb…”

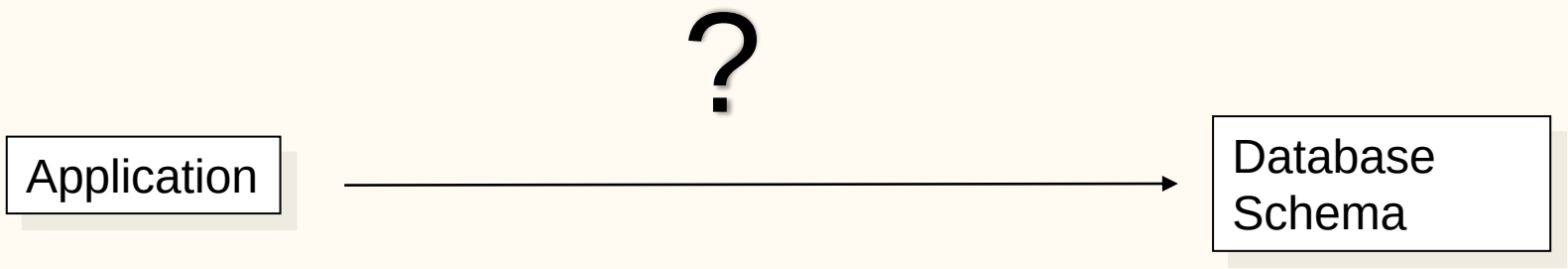
The Application

"An author can write zero or more books and a book is written by one or more authors. Where a book is uniquely identified by its book-id. For each book, we also record its title, price, and availability. An editor is uniquely identified by his/her reader-id and we also record his/her name, phone number, address. The address is composed of street and suburb…"

This is the application we need to design the design for; the application has a set of requirements.

Simplified Database Workflow

What should we do?

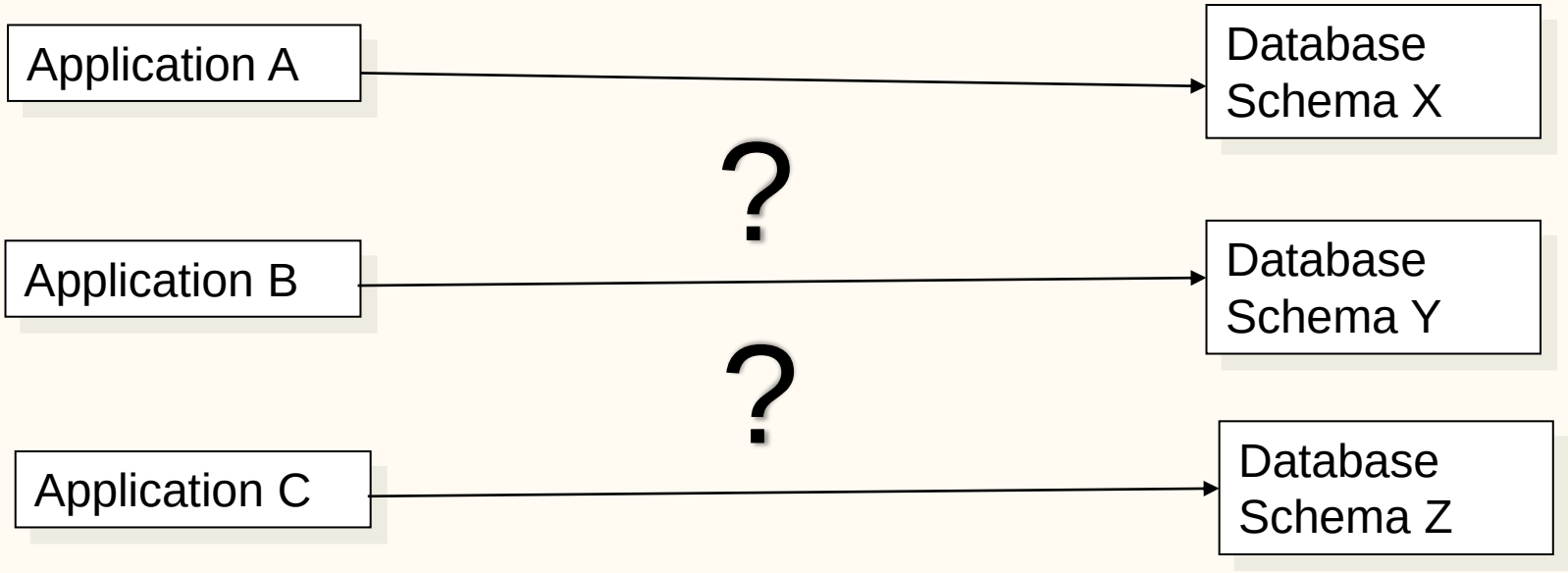


Application: has many requirements

Database Schema: provides a logical view of data model

Simplified Database Workflow

More Generally, how should everyone build his/her databases for an application for its given requirements?



Database Schema: provides a logical view of data model

Solution

Dr. Peter Chen

https://www.csc.lsu.edu/~chen/

His work started a new field of research and practice:

Conceptual Modeling

See paper on the Entity-Relationship Model in link

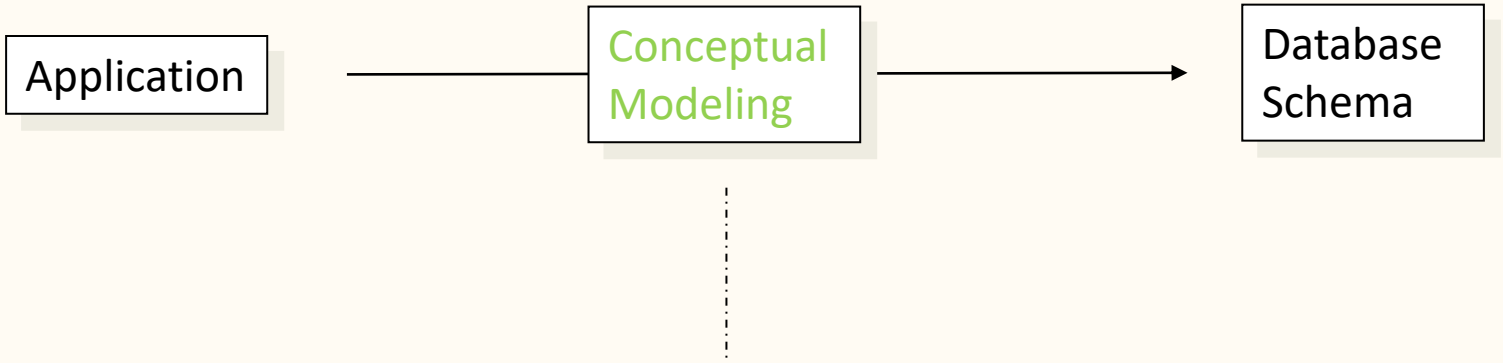
https://bit.csc.lsu.edu/~chen/pdf/english.pdf

English Sentence Structure and Entity-Relationship Diagrams

PETER PIN-SHAN CHEN*

Graduate School of Management, University of California, Los Angeles, California 90024

Solution



(Chen. 1976)

- Entities
- Relationships
- Attributes

Entity-Relationship Model

Chens ER model has two major components:

- ***Entity***: collection of attributes describing object of interest
- ***Relationship***: association between entities (objects)

There's also the third components unofficially

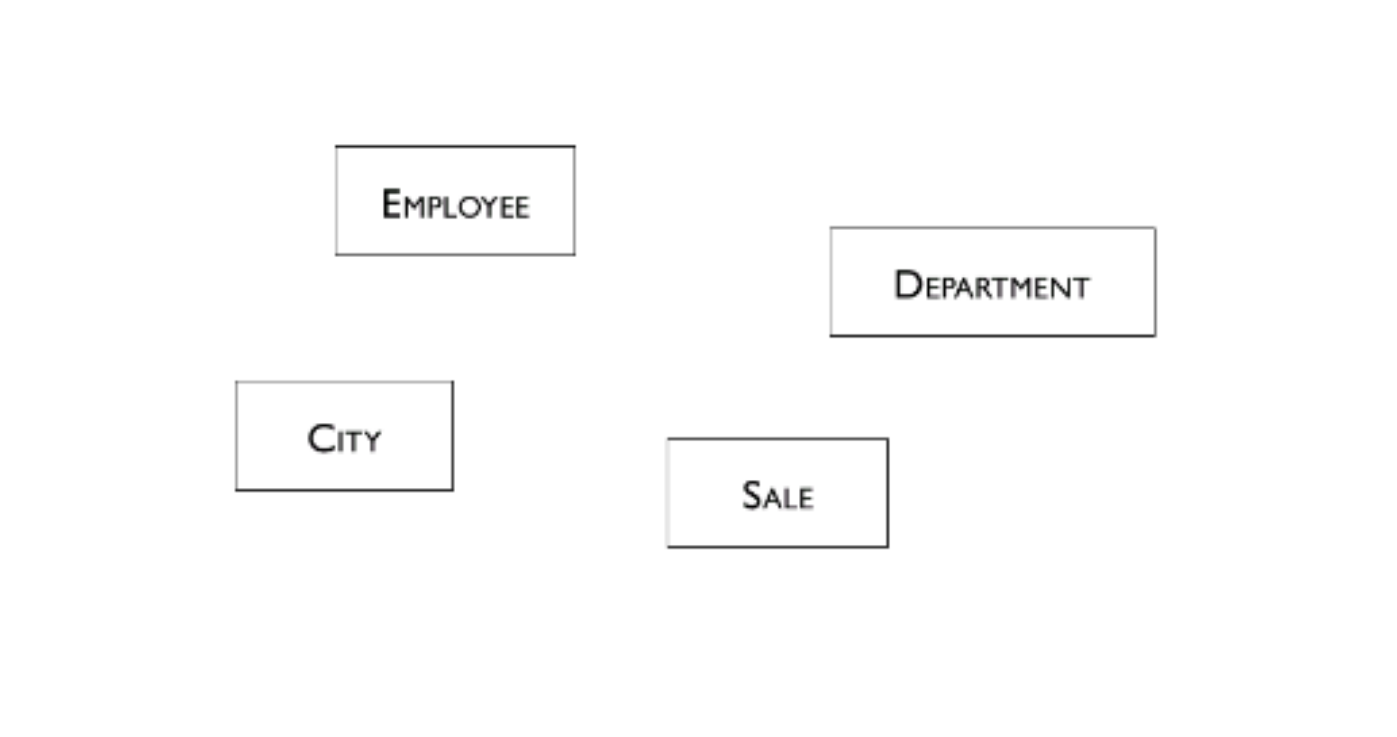
- ***Attribute***: data item describing a property of interest

Entity-Relationship Model

“Entities represent things in the real world, and attributes describe properties of entities.”

Entities

Some examples of entities



Simple Attributes

Attributes may be **simple (atomic).**

Each simple attribute of an entity type is

associated with a **value set** (or **domain** of values), which specifies the set of values that may be assigned to that attribute for each individual entity.

For example:

- Entity = Student
- Attributes = Student number, name…

Student A:

- Student number = z12345678
- Name = Bob
- …

Entity-Relationship Model

Attributes can also be multi-valued

- Multivalued: has more than one value
- No longer a simple attribute with only one value.

Questions:

- Why do we need multi-value attributes?
- Can’t we do everything with simple attributes?

Problem

Each simple attribute is meant to hold one value, sure it can change… but one value at a time.

Questions:

- What if… I'm asking you to model shirts with two colours?
- How can we model this without multivalued attributes?

Problem

Do we need multi-value attributes?

The attributes you design should be able to describe, and the combinations of all its instances should be able to express the entity faithfully

Example: more expressively model the attribute colour for entity shirt.

Composite Attributes

What is a simple attribute? Attributes that are not divisible are called *simple* or *atomic attributes*.

Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings.

Some semantics cannot be captured using atomic attributes

Question:

Question: is *Address* a simple attribute/value?

- *Address = 'Computer Science Building (K17), Engineering Rd, UNSW Sydney, Kensington NSW 2052"*

How can should we model Addresses ?

Question:

Question: is *Address* a simple attribute or a composite attribute?

- *Address = 'Computer Science Building (K17), Engineering Rd, UNSW Sydney, Kensington NSW 2052"*

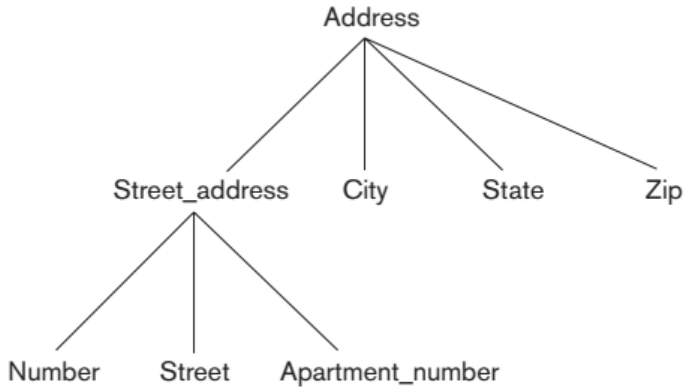


Figure 7.4
A hierarchy of composite attributes.

We also avoided this possible situation:

- Address = "that red house on parks street"

Question:

Composite attributes are useful for situations when

- The end-user sometimes refers to the composite attribute as a unit,
- But at other times refers specifically to its components.

Question: Can’t I just let my composite attributes, be simple attributes instead?

Derived Attributes

Attributes that are problematic if modeled with a simple value.

Scenario: modelling a person's age:

Why? Your values can change and are dynamic.

The age attribute is called a **derived attribute**, because we is said to be **derivable from** the Date-of-birth attribute, which is called a **stored attribute**.

Derived Attributes

Where there is a derived attribute, there must also be an attribute where it's values can be derived from.

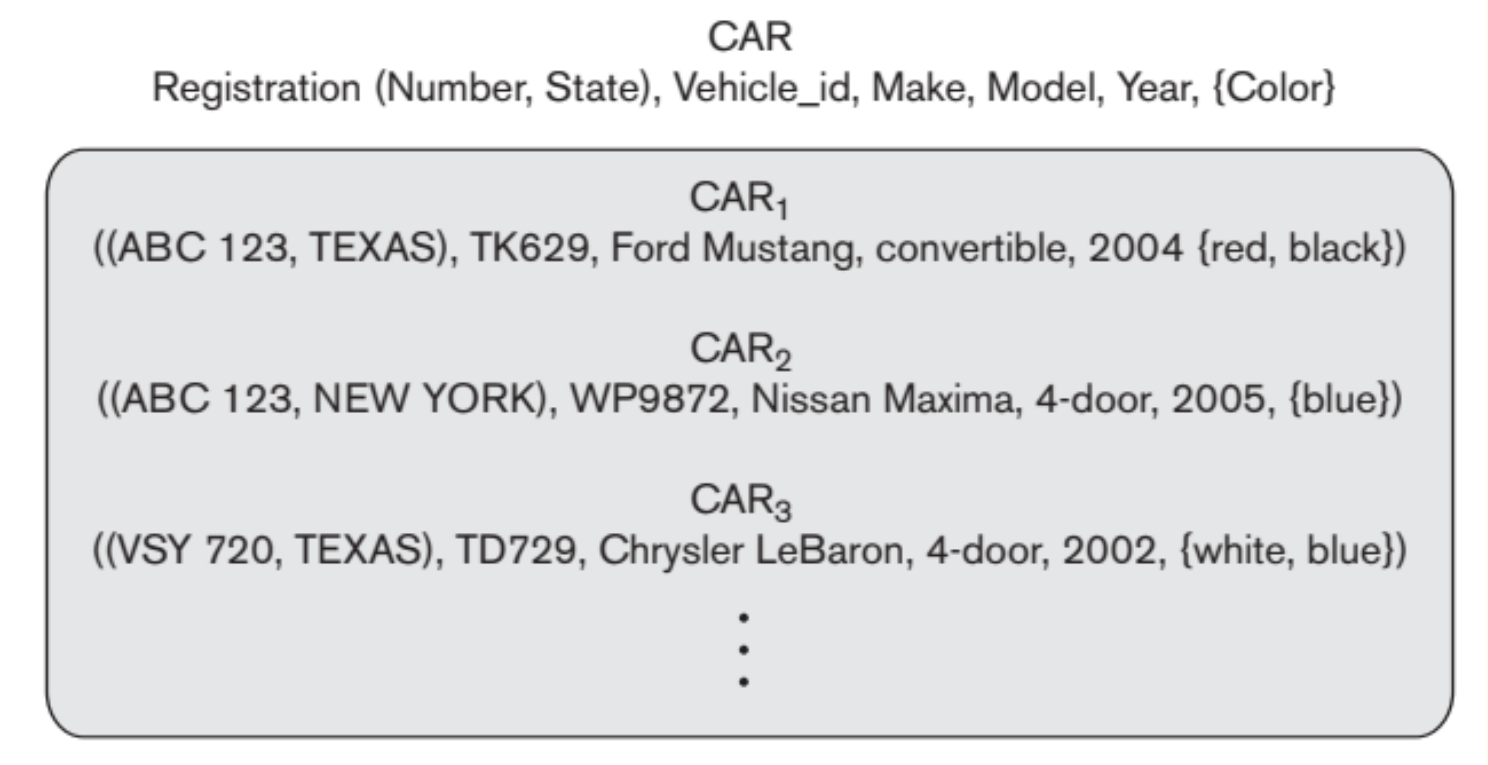
Derived attribute values are not stored, the stored attribute that derives the value is stored.

Question: **Why include it at all if it's derived? Why not leave it out completely in the data model?**

Take away

1. An entity type describes the **schema** or **intension** for a *set of entities* that share the same structure.
2. An entity type is represented in ER diagrams as a rectangular box enclosing the entity type name.
3. The collection of entities of a particular entity type is grouped into an entity set, which is also called the **extension** of the entity type.

Take away


CAR
Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}
CAR1
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})
CAR2
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})
CAR3
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

Take away

- Given an entity, the attributes you design should be able to describe, and the combinations of all its instances should be able to express the entity faithfully.
- This is why attribute types such as multi-labelled attributes, composite attributes exist.
- Can I have an entity to describe something abstract? Of course, entities don't have to concrete/ tangible objects. You will still need to find the right attributes to describe it.
- Relations also have attributes (spoilers)

Entity Instances/Facts

A good entity schema should have good attributes that can be filled with good values.

Car Entity Schema:

- ((Reg. Num, State), Make, Model, Year, {Colour})

Car instance:

- (CS9311, NSW), Toyota, Toyota Corolla, 1999, {White, Silver})

NULL (1)

What if an instance doesn't have a value?

Examples:

- Marriage date of a person is not known (True value is not yet known)
- Student may have not had a pet (No suitable value)

Exercise: List more cases where use of a NULL value would be appropriate.

NULL values represent attributes whose values are unknown or do not exist for some entity instance. In general, it is a special value to indicate a lack of value.

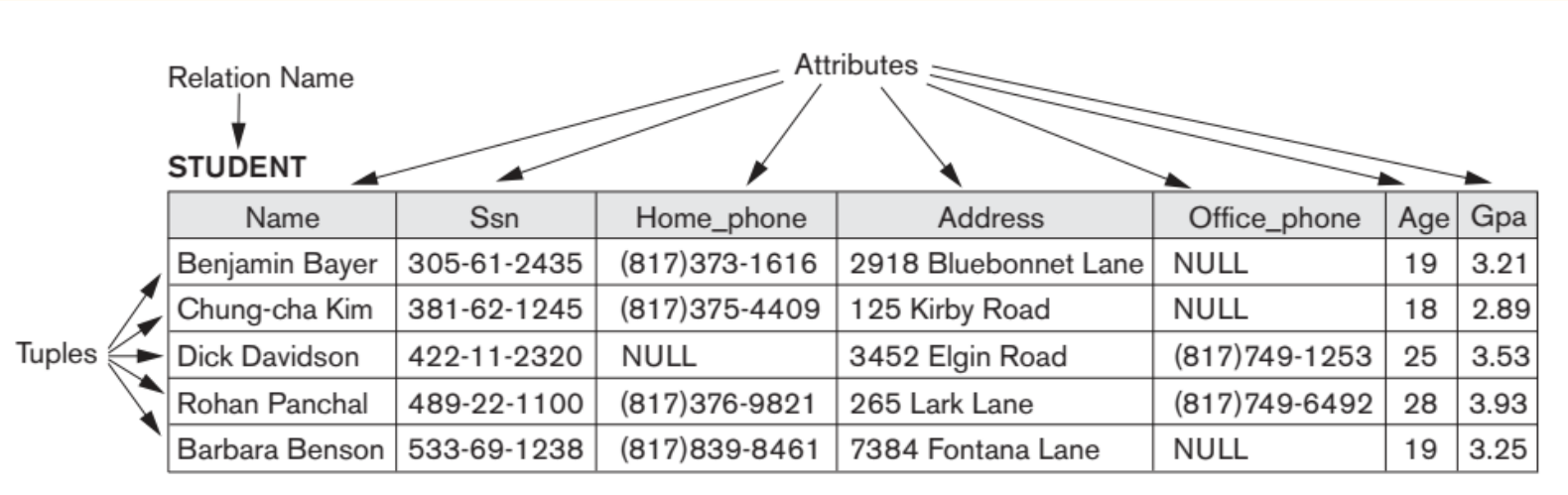
NULL (2)

Null: Example Usage: COMP9311 Student Schema:

- (zID, trimester id, ass1-mark, ass2-mark, exam-mark)

COMP9311 Student Instance/Fact

- 21T2 = (z0000001, 21T2, 16/25, 18/25, 50/50)
- What about someone from this 22T1? use NULL and update it later?



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Keys

Key constraint in ER Modelling: in any extension of the entity type, no two entities have the same values for their key attribute or key attributes.

For example:

- {payroll number} is a key of RESEARCHER,
- {car registration number} is a key of CAR.
- Person (sex, name) will have instances: (Female, 23); (Male, 34); (Female, 23)

All elements of a set are distinct All elements of a set are distinct; an entity can be defined as the set of its entity instances.

Therefore, the entity instances in a relation must also be distinct. How do we identify them? There must be a key for all entities.

Key

A ***super key*** is a set of one or more attributes that uniquely identify any entity instance of an entity type.

- natural (e.g., name + address + birthday)

Q: What if none of my attributes can uniquely identify my entity?

A: you can make an attribute artificially (e.g., SSN)

Key

In real-life, there is likely to very be than one valid possible key…

See schema for *Person (DNA Sequence, Passport Number)*

Keys containing multiple attributes as **composite keys.**

Composite key aka **compound key**.

Key

Q: Technically, can't I make a entity key to be all its attributes?

A: Since the entity is a **set**, in the worst case where there is no natural composite key, the set of all it's attributes together should uniquely identify its values (strictly speaking).

Every entity must have a final primary key: a minimal set of attributes that can uniquely identity its entity instances.

Key Summary

Key/ Super key: a set of one or more attributes that can uniquely identify an entity instance of an entity type.

Candidate key: a key chosen by DB designer

Primary key : a minimal candidate key. A primary key is a super key and a candidate key

Weak Entity

Weak entity, entities with the property that they can’t have a natural primary key of their own.

Scenario: Course and Section

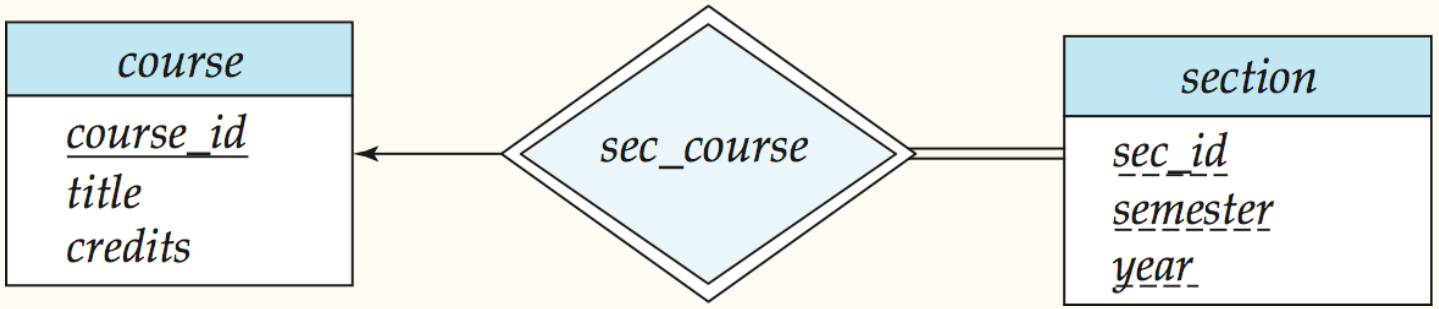
Weak Entity Sets?

All entity sets have a primary key. They exist because they are independent.

There does not exist a primary key in “section”.

- there may be two, (1, Spring, 2010) and (1, Spring, 2010).
- They are two distinct entities from different courses, but we cannot manage them.

Consequently, the section is not an independent entity set, because it’s there's no primary key. But the section should be entity (we need it to be). In that sense, it should be something weaker.

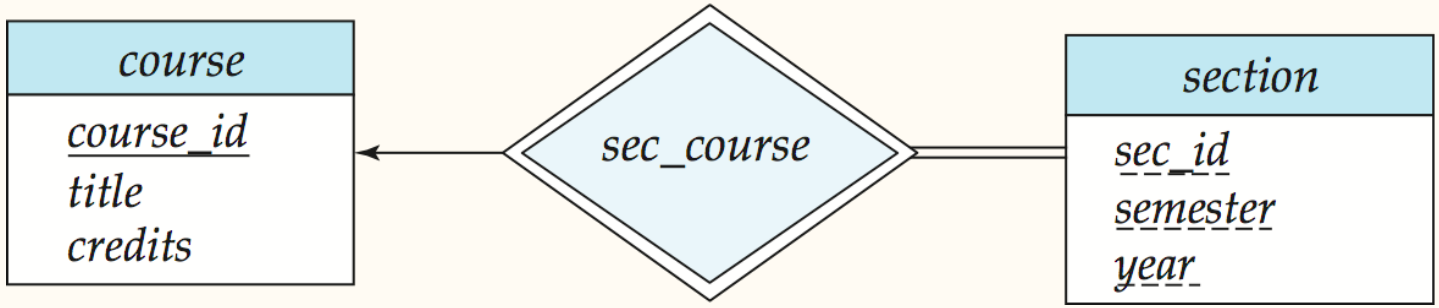


Make Weaker Stronger?

What if… How about we make sec_id unique in the section entity set?

Add an artificial primary key sec_id and section becomes an entity set or precisely a strong entity set.

Could this work?



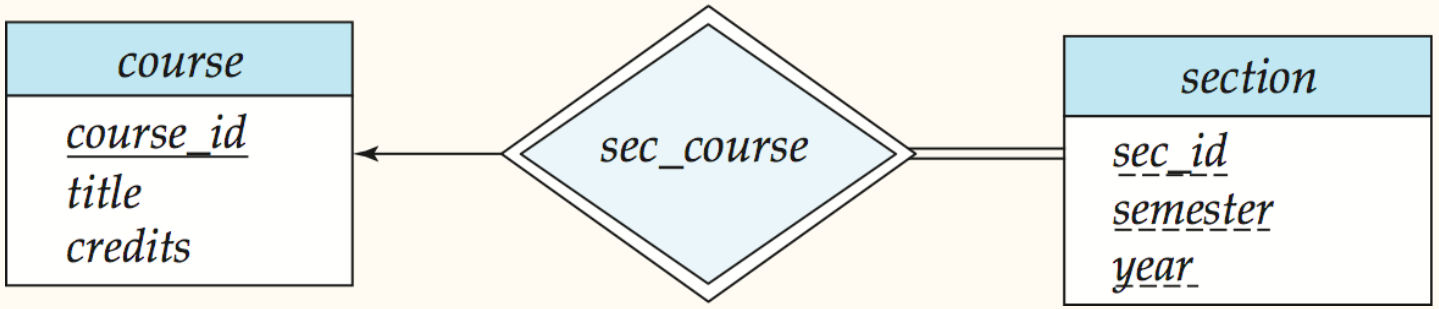
Make Weaker Stronger?

But conceptually, a section is still dependent on a course for its existence.

How about we add a course_id into the section entity set?

- Make it Section has a primary key (course_id, sec_id, semester, year).
- Section becomes an entity set or precisely a strong entity set!

Will this work?



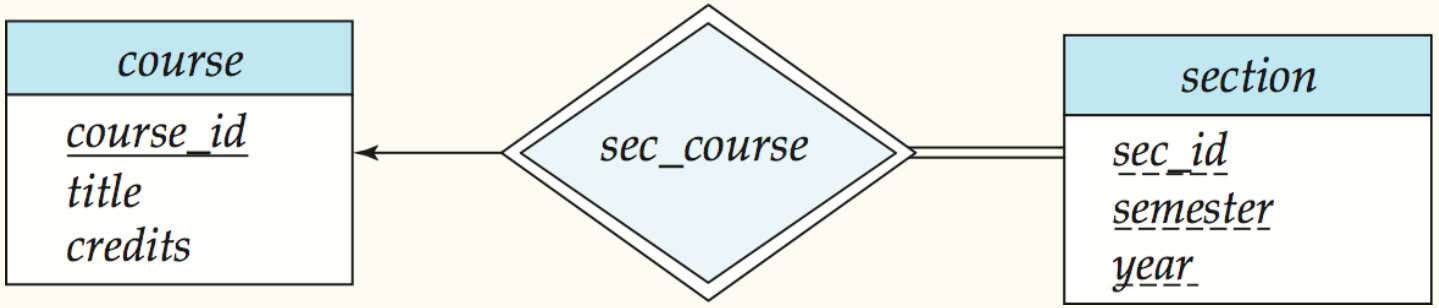
Make Weaker Stronger?

There are a few big problems:

- course_id is not a part of section, esp. since we treat section as an entity set.
- course_id is to identify the course entity set not the section.

And what about the preexisting relation?

- (1) if we add course_id into the section entity, the sec_course relationship becomes redundant. (2) But, without sec_course, the relationship between course and section becomes implicit, which is not what we want.



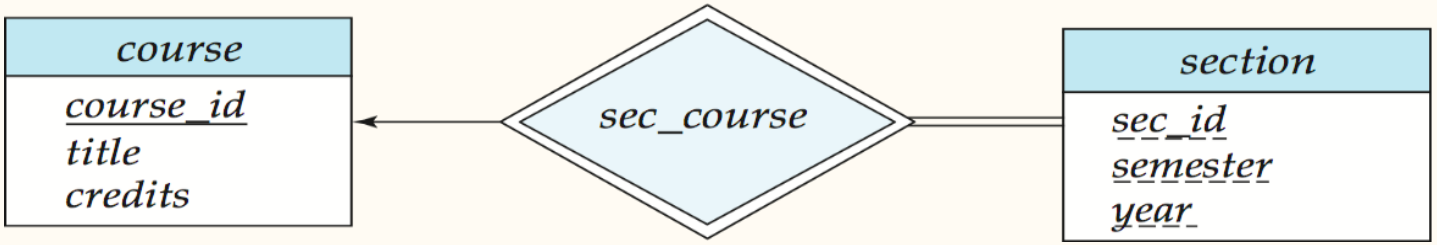
But we need a primary key!

How about we borrow the course_id from the course entity set.

The primary key of the section is formed by

- (1) primary key of the course on which the section is existence dependent
- AND (2) the weak entity set's discriminator sec_id.

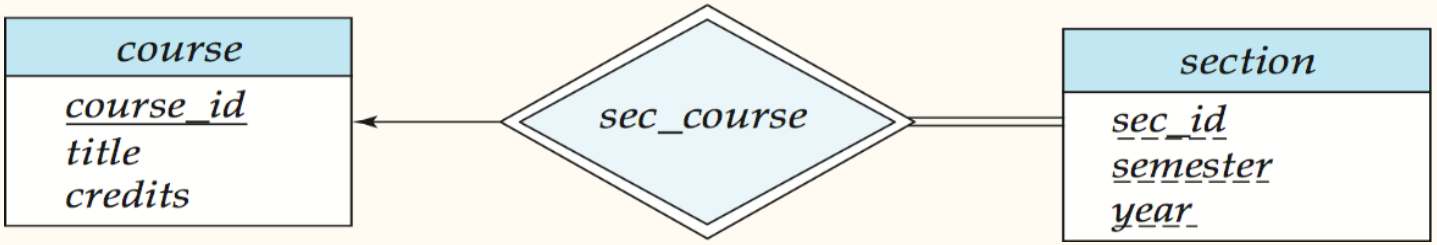
The discriminator (or partial key) of the section is the set of attributes: sec_id, semester, and year.



But we need a primary key!

The existence of a weak entity set depends on the existence of an identifying entity set

- It must relate to the identifying entity set via a total; one-to-many relationship set from the identifying to the weak entity set
- Identifying relationship is depicted using a double diamond

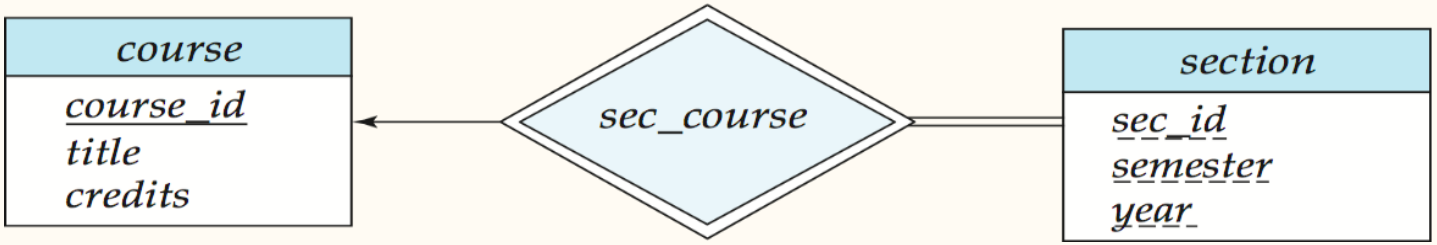


Weak Entity Sets (1)

An entity set that does not have a primary key is referred to as a weak entity set.

The existence of a weak entity set depends on the existence of an identifying entity set

- It must relate to the identifying entity set via a total; one-to-many relationship set from the identifying to the weak entity set
- Identifying relationship is depicted using a double diamond



Weak Entity Sets (2)

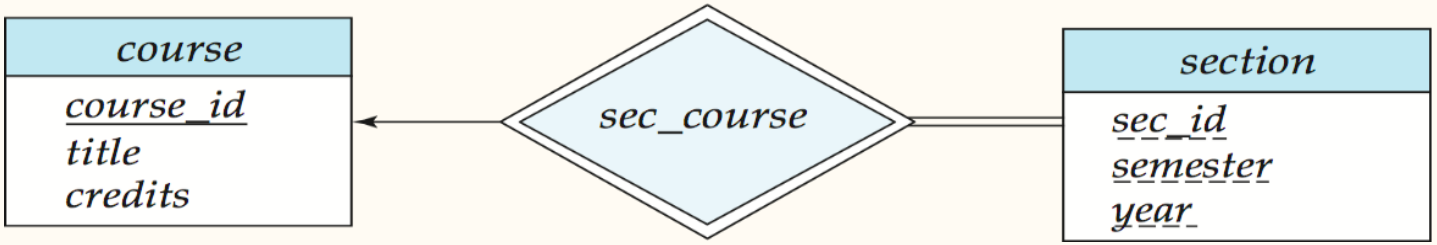
We underline the discriminator of a weak entity set with a dashed line.

We put the identifying relationship of a weak entity in a double diamond.

Primary key for section:

(course_id, sec_id, semester, year)

Note: the primary key of the strong entity set is not explicitly stored with the weak entity set. It is implicit in the identifying relationship.



Summary

A weak entity doesn't have any primary key but does have a partial (partial discriminator) key.

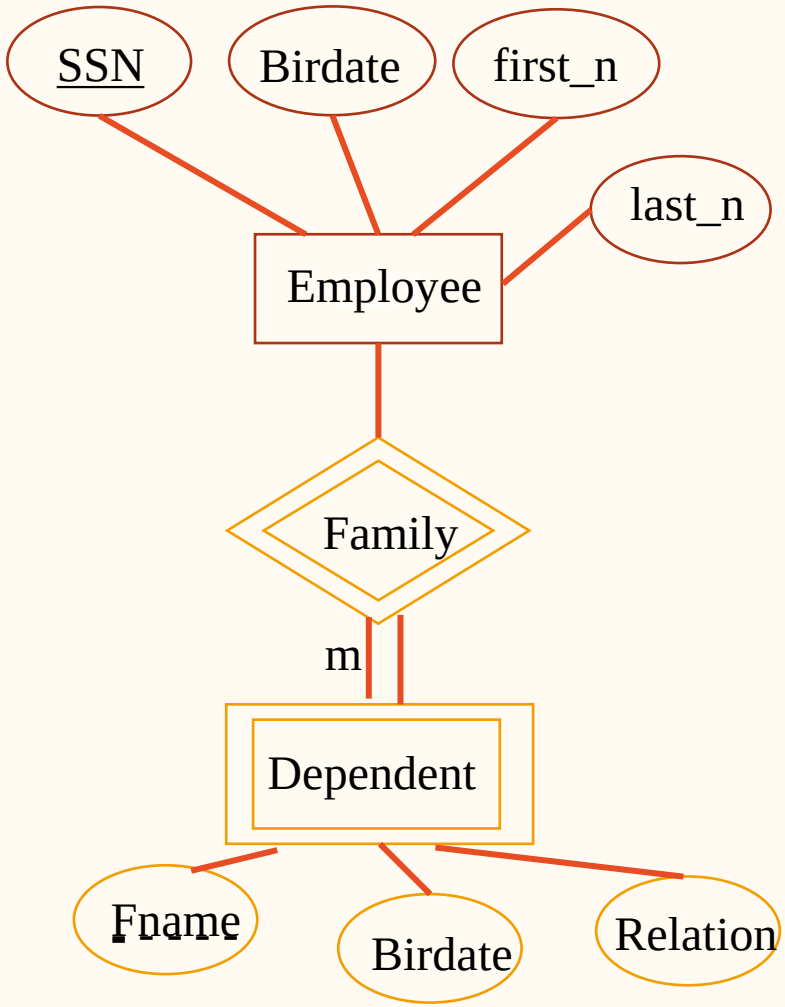
Partial discriminator key: a key that is partially unique. i.e., only a subset of the attributes can be identified using it

A ***strong entity*** is an entity a regular entity where there is a primary key that uniquely identifies all instances.

Identifying relationship: the relationship type between a weak entity type to the owner of the weak entity type.

Example

1. A TAX PAYER entity may be related to several DEPENDENTS, identified by their names.
2. DEPENDENT is called a weak entity, {Name} is a partial key for it.
3. The identifying relationship between DEPENDENT and TAX PAYER is IS-DEPENDENT-OF.
4. TAX PAYER is said to own DEPENDENT.



Database Development Workflow

Nowadays, database design process is a mature one.

- Analyse requirements and develop a high level model.
- Implement the data model as relational schema.
- Implement the relational schema as a database schema.

Visualizing an ER Data Model

At a daily meeting:

A: “so far we completed that high level data model that you wanted sir…”

A: *hands a piece of paper*

Paper reads:

“There is one strong entity type that is a Car.

It has a multi-labelled attribute to describe its colour.

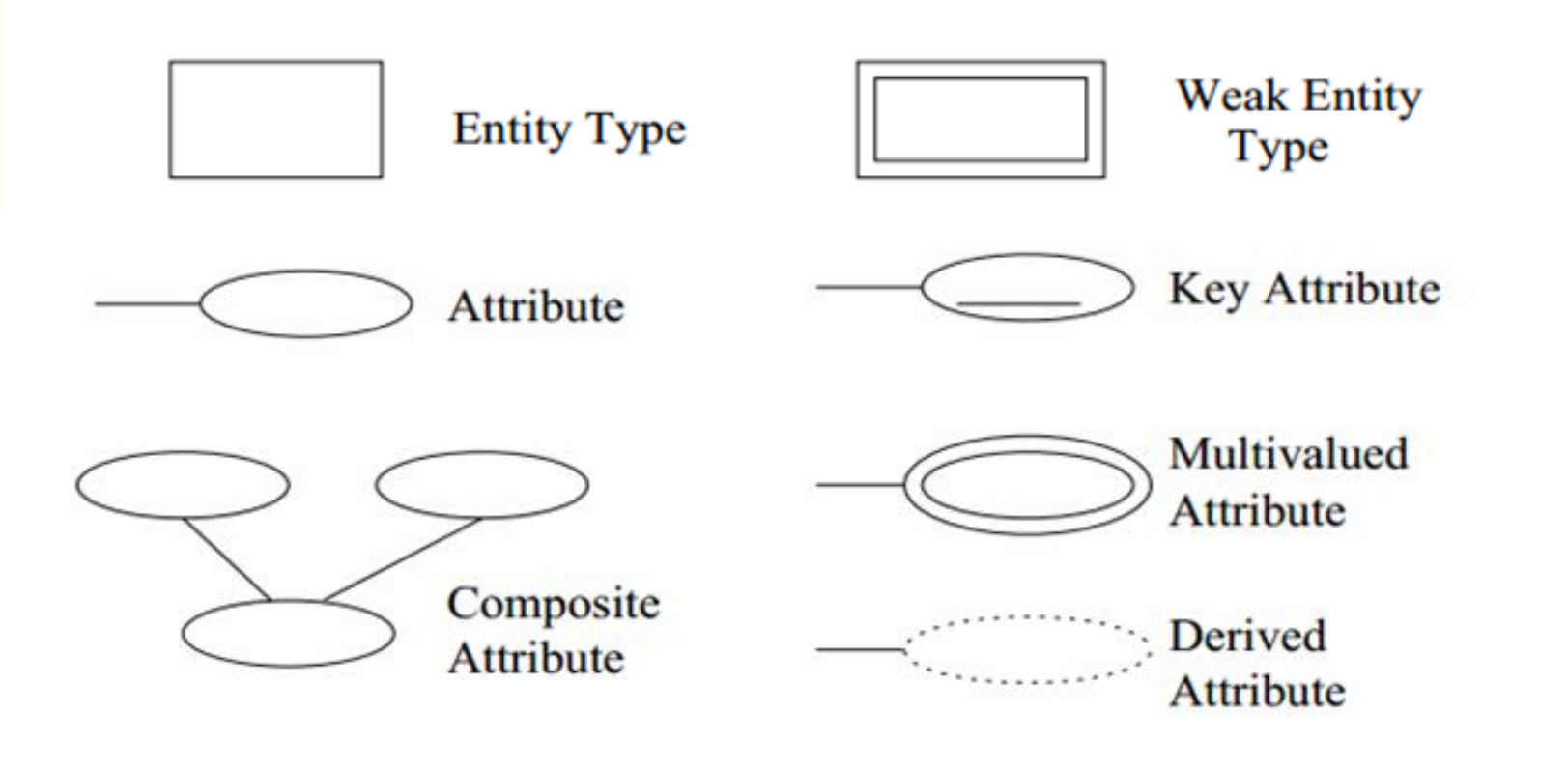
It has Registration with is made of Registration Number and State.

It has a few more attributes Make, Model and Year

…”

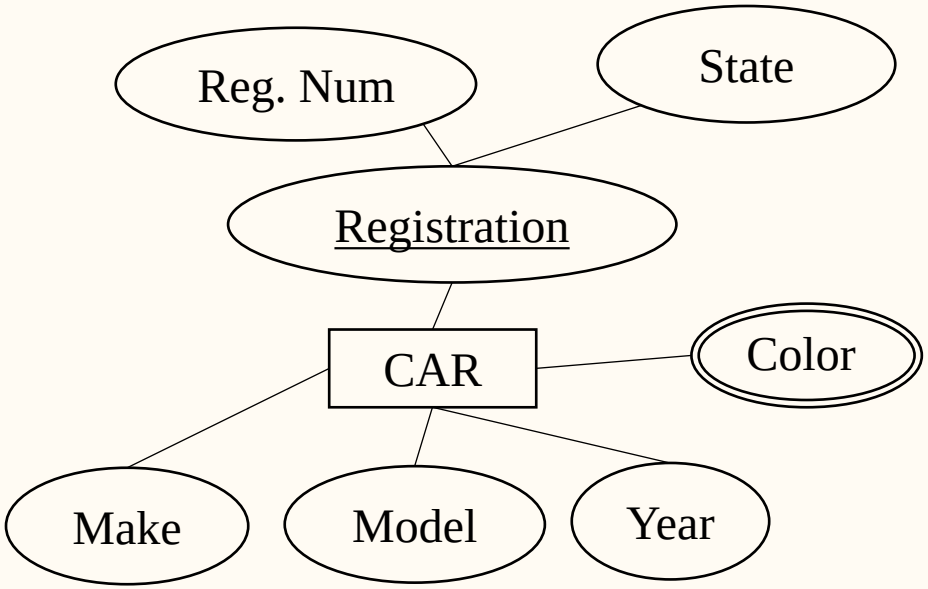
Visualizing an ER Data Model

Notations:



Visualizing an ER Data Model

The data model can be effectively described using the Entity-Relationship Diagrams (ERDs).



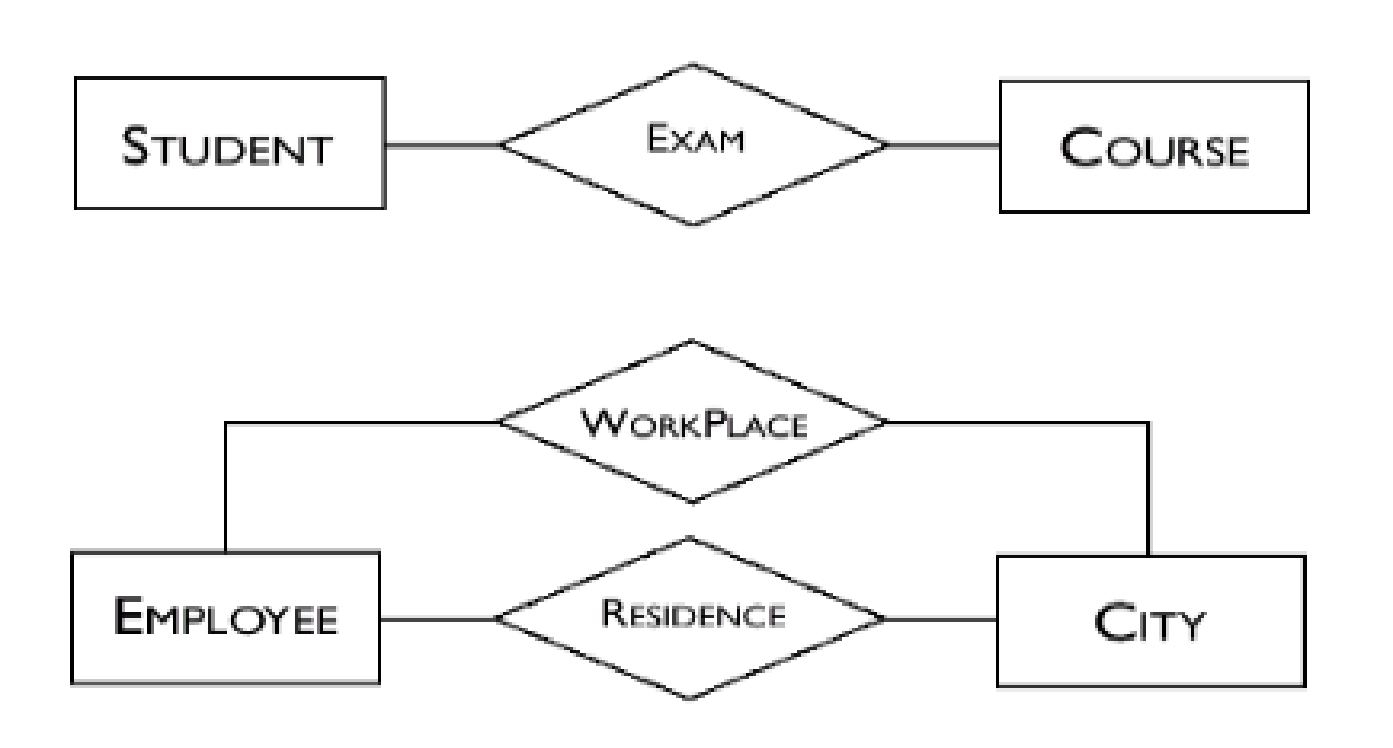
Break 10 min

Your lecturer recently realized 2 hours of talking wasn't as easy as 2 hours of listening.

Next: Relationships

Relationship

Second Big Component of ER: They represent logical links between two or more entities.



Relationships

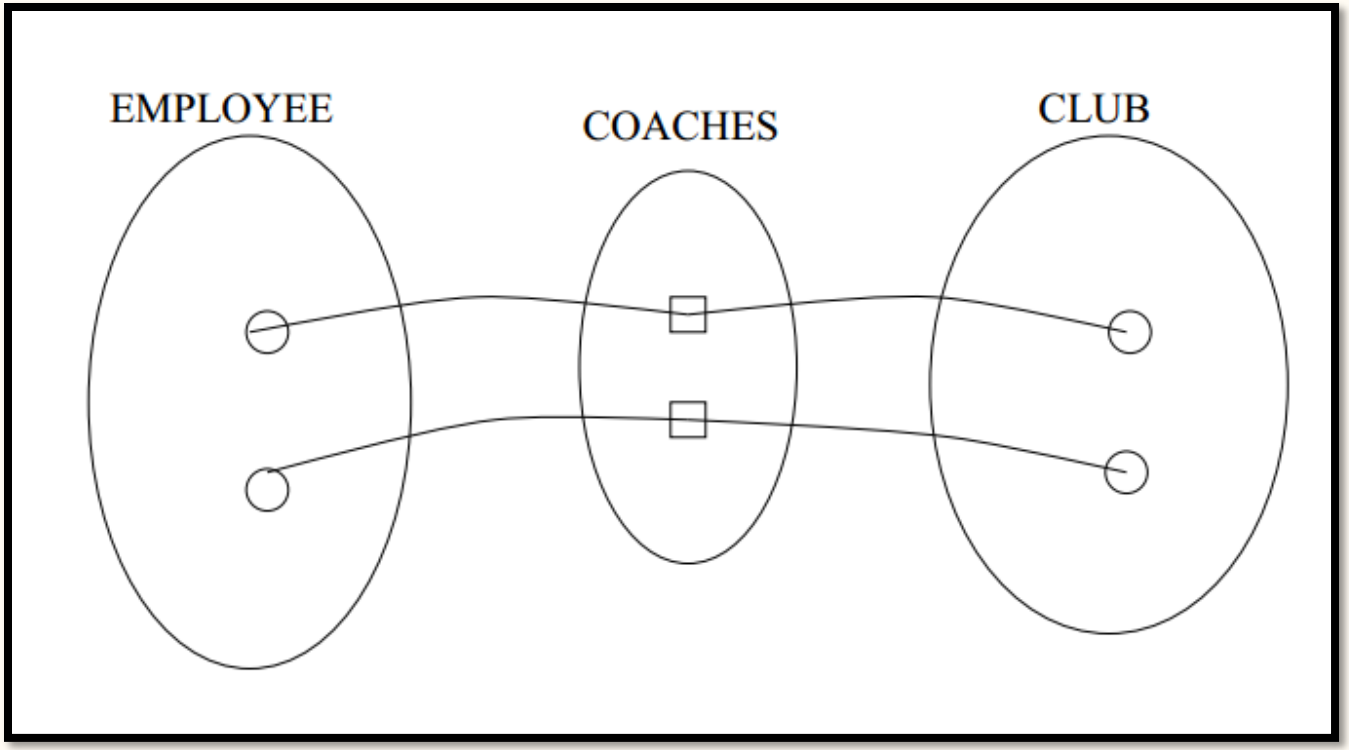
Relationships relates one entity type to another entity type.

An entity type can be related to more than one other entities, and will have a different role to play for each relationship (Name each relationship to distinguish them).

An entity can also have different relationships with another entity.

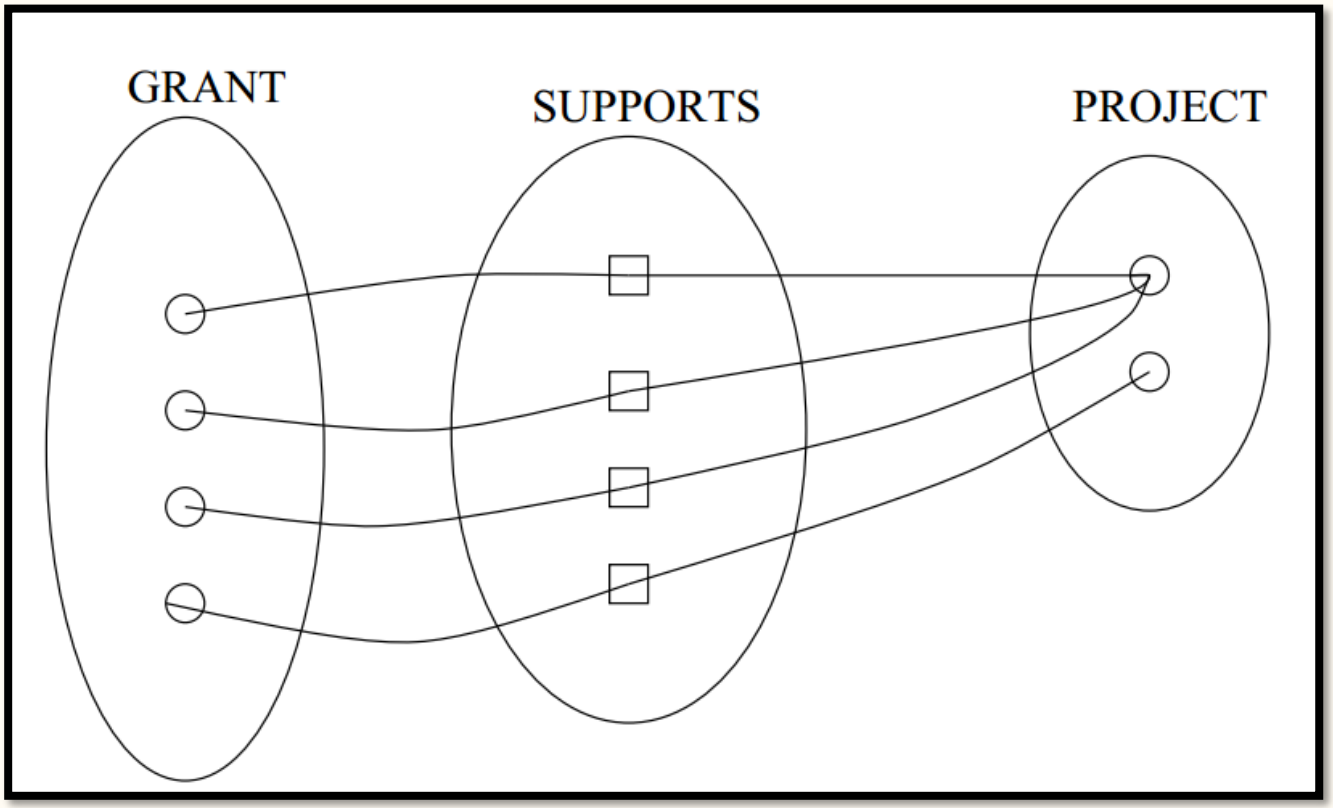
Practice

Coaches is a relationship. It's instances describes the relation between a coach and a club.



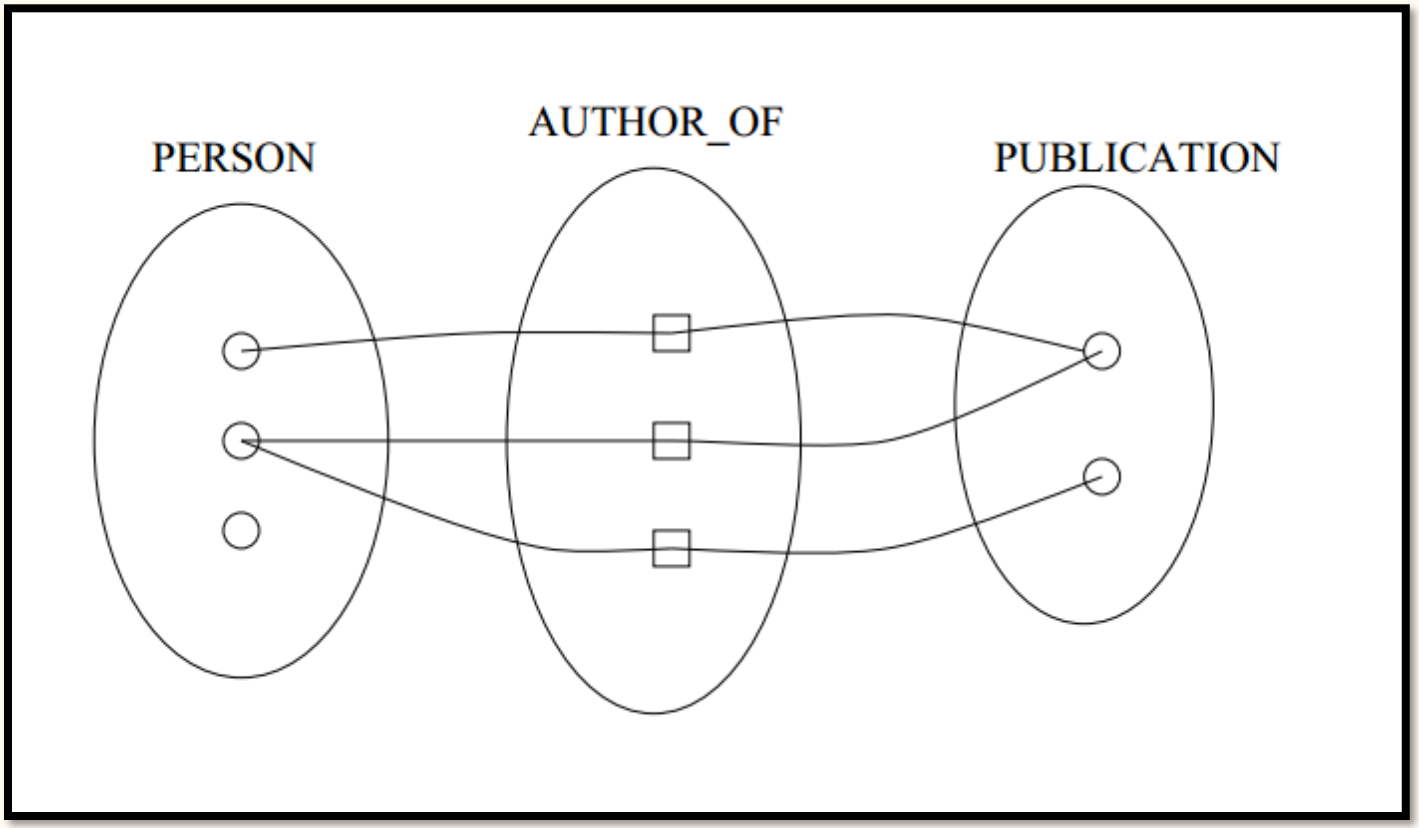
Practice

Supports is a relationship. It's instances describe the how grants are given to projects.



Practice

Author-of is a relationship. It's instances describe who wrote what publication.



Modelling relationships

Relationship types usually have certain properties that limit the possible combinations of entities participating in relationship instances.

By default, entities can be related to other entities freely.

Are all relationships like that?

1. Relationship between employee and club?
2. Relationship between moon and planet?

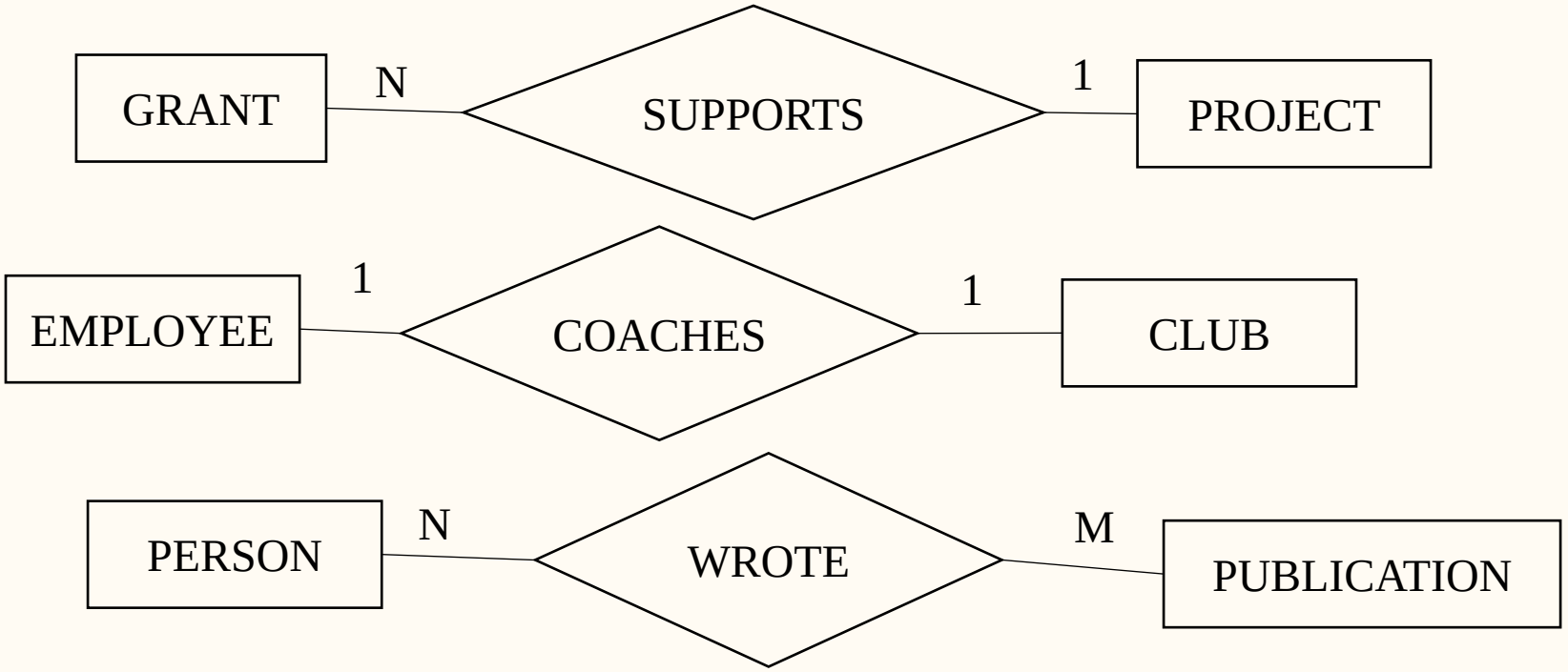
Modelling relationships

Cardinality ratio: the number of relationship instances an entity can participate in.

There are three types of cardinality in relationship: is M:N (many to many), 1:N (one to many) is stricter, and 1:1 (one to one) is the strictest.

Constraints on relationship types(cont)

We can also show this in an ERD:



Relationship properties

Will each instance participate in this relationship?

Relationships with a special property to be expressed.

For example, all university students must be enrolled to university/universities.

So far, entities don’t have to participate in the relationships they are in.

By default, participation is ***partial***

Constraints on relationship types

Total participation: each entity instance must patriciate in at least one relationship instance.

Example: We want this relationship to express all publications must be written by a person.

Constraints on relationship types

e.g. every publication has an author.

There can’t be a publication without an author

e.g. not every person has publications.

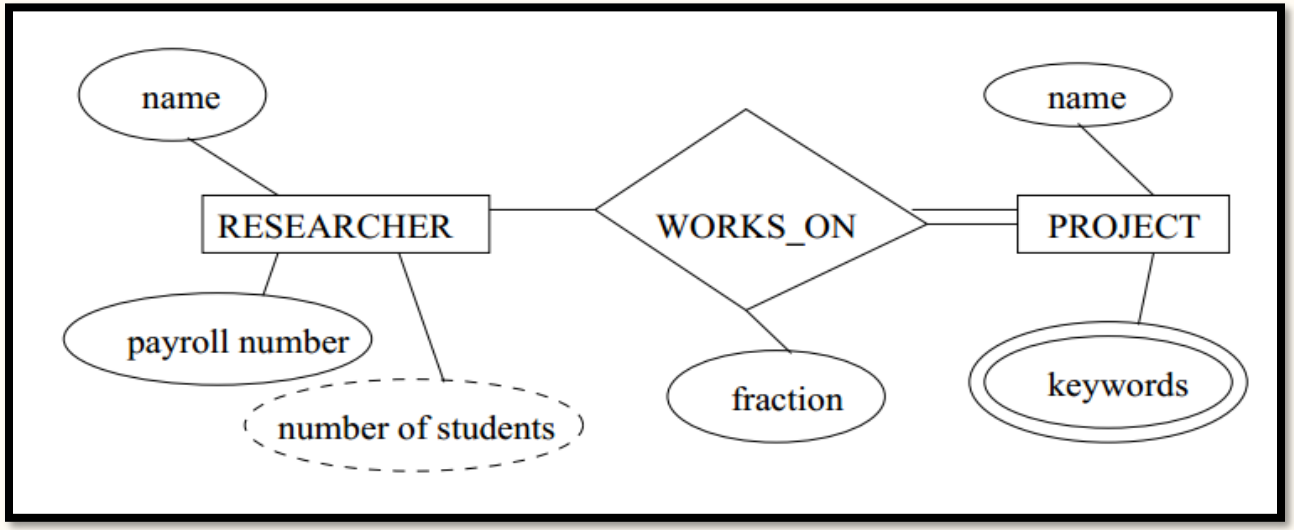
There can be people who don’t write publications

Attributes (Relationship)

A researcher may work on several projects.

The fraction of her time devoted to a particular project could be an attribute of the WORKS ON relationship type.

Q: Why not put the attribute on either researcher or project?



More on Relationships

Binary relationship: relationship of degree 2

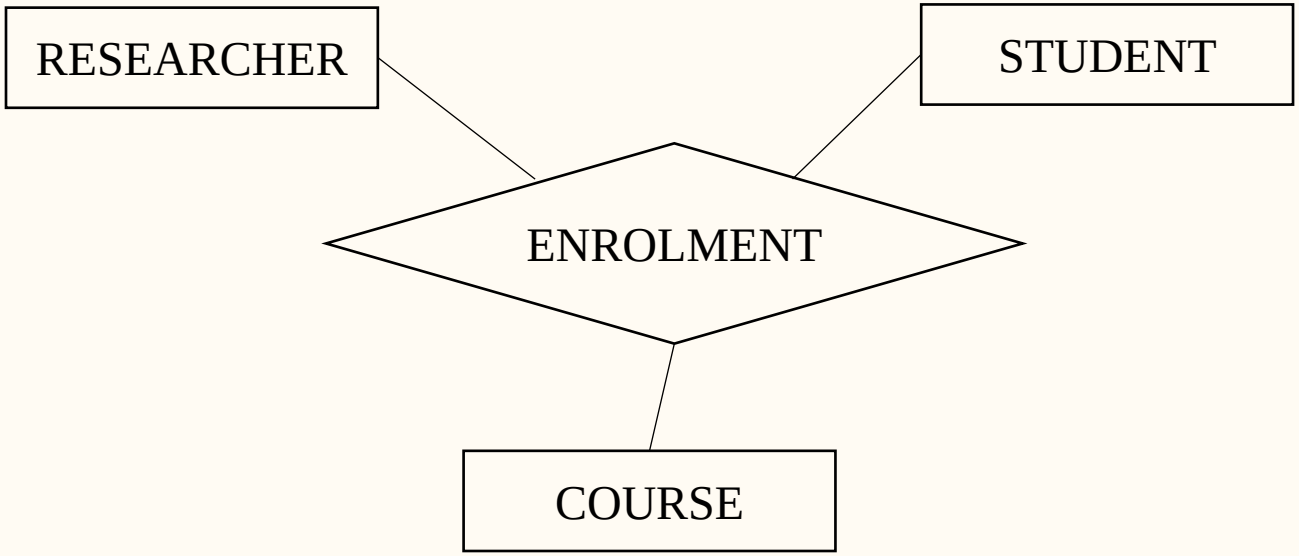
Degree (of a relationship): number of participating entity types.

Lectured cover binary relationships.

More on Relationships

Consider the setting: relationship between RESEARCHER, STUDENT and COURSE.

ENROLMENT could be a ternary relationship (degree 3)



On the Degree of a Relationship:

The more entities in a relationship, the more careful you must be when describing it. What you think your expressing may not be what the diagram means exactly.

Rule of Thumb as you are starting out: If you think you have a ternary (or higher) relationship, decompose it to binary relationships.

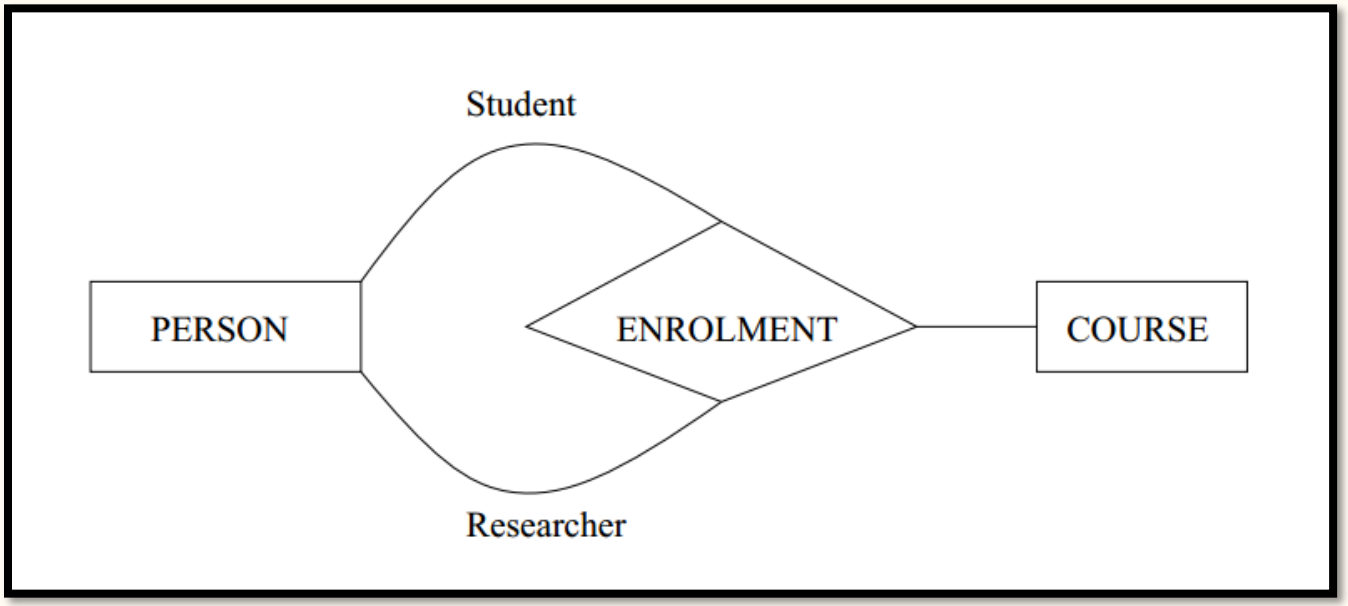
Recursive Relationships

Relationships involving the same entity is possible, they are known as recursive relationships (aka unary relationships).

Exercise: Think of possible degree 1 relationships

Backtrack: ENROLMENT

To make things interesting, this is another way of modelled the former.



(Part of this is a recursive relationship/unary relationship)

Five Minute Break

Digest contents thus far.

Question (1)

Q: **Is a Car… an Entity or a Relationship?**

Question (2)

Q: **Are Entities and Relationships equally important in the ER model?**

Question (3)

Q: **Should I model a bus waiting behind a car as a relationship?**

Technically yes. If it must be the case.

Think ahead, what will you be keeping in the database if relationships are temporary interactions?

Inherent long-term interactions are more logical.

Rule of Thumb

Keywords in requirements suggest data/relationships

(rule-of-thumb: nouns → data, verbs → relationships) but it depends…

Consider the following while we work through exercises:

Start simple ... evolve design as problem better understood

Ockham's razor: simple is best. Don’t try to complicate it

Identify objects (and their properties) first, then relationships

Conceptual Data Modelling

1. Picking the right kind of entity: does it have significant properties and describe objects with independent instances?
2. Picking the relevant relationships: does it provide a logical link between two (or more) entities?

There is no best data model for an application. However, the important aspects of a design (data model) are:

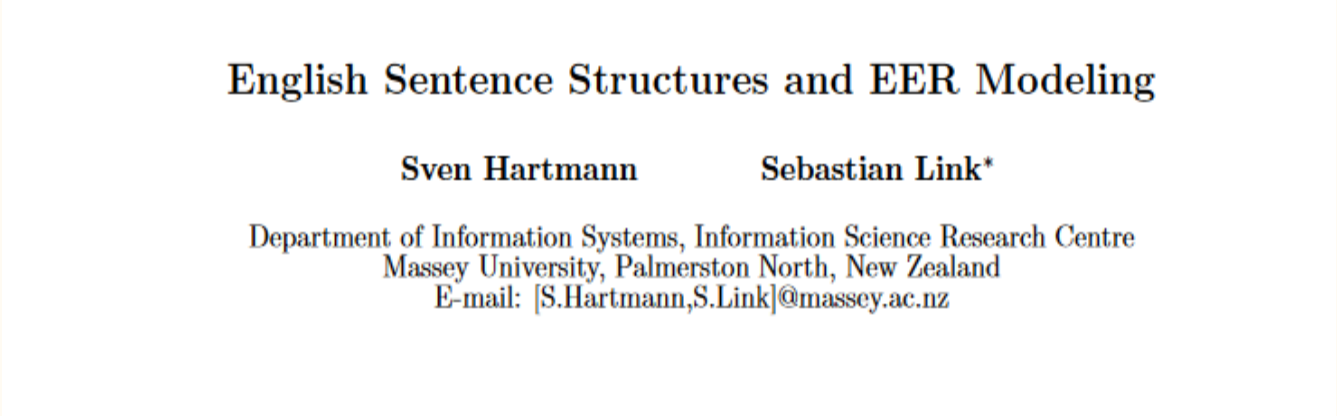
- Correctness: w.r.t the ER model
- Faithfulness and Completeness : w.r.t the requirements
- Minimize redundancy (Remember which attribute serves this purpose?)
- Readability

That's the Entity Relation Model

1. ER models gave us the first rules to capture the semantics
2. Systematically change it to Relational logical models (spoilers)
3. Allowed a lot of people to take a step from 'requirements from applications' to 'implementing the database'

Enhanced ER (EER) model(cont)

In time, improvements were made to ER model.



English Sentence Structures and EER Modeling

Sven Hartmann Sebastian Link*

Department of Information Systems, Information Science Research Centre
Massey University, Palmerston North, New Zealand
E-mail: [S.Hartmann,S.Link]@massey.ac.nz

We will look at one:

- *Specialisation*: the process of defining a set of subclasses of an entity type; this entity type is called the superclass of the specialization.
- *Generalisation*: a reverse process of specialisation.

Enhanced ER (EER) model(cont)

A specialisation involves the following aspects:

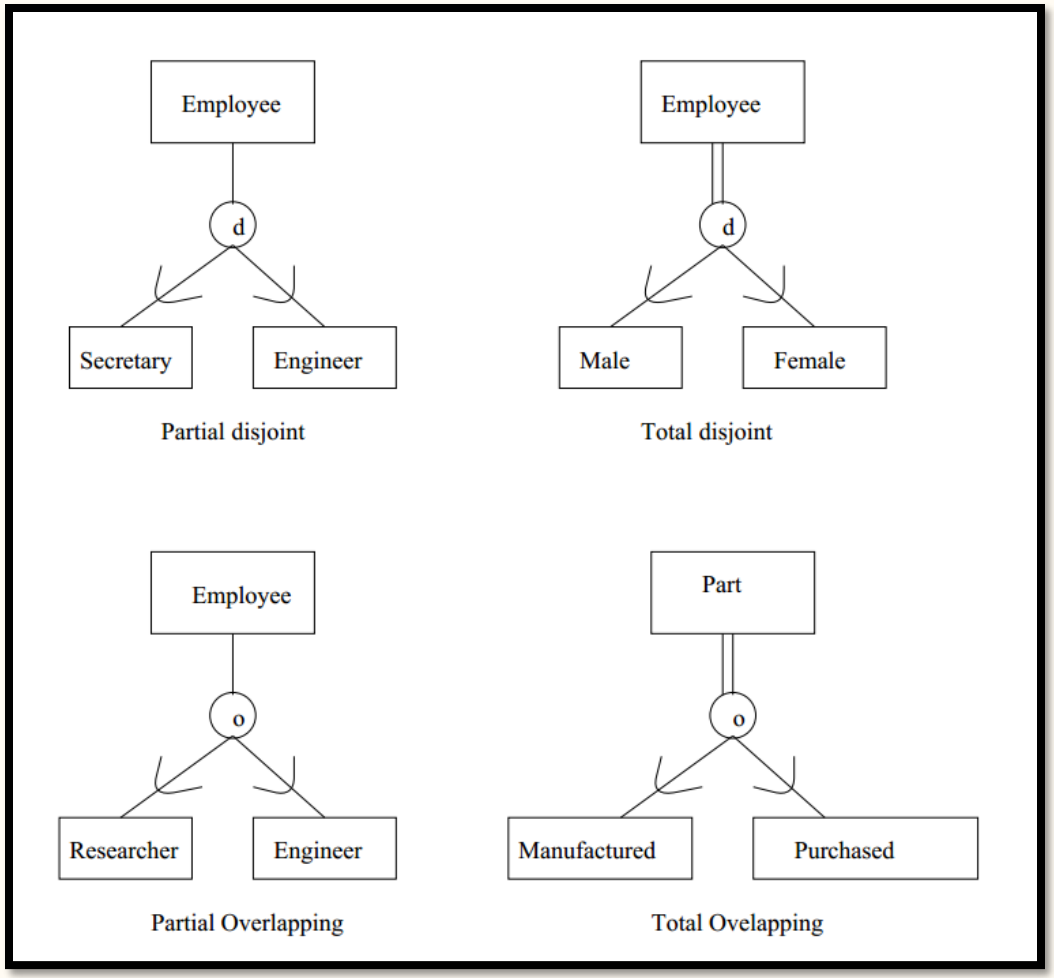
- Define a set of ***subclasses*** of an entity type.
- Associate additional specific attributes with each subclass.
- Establish additional specific relationship types between each subclass and other entity types, or other subclasses.

A subclass may have multiple superclasses.

A specialisation:

- may be either total or partial; and
- may be either disjoint or overlapping.

Enhanced ER (EER) model(cont)



Final Exercise: A Library Database

- A book is uniquely identified by its book-id. For each book, we also record its title, price, and availability.
- A reader is uniquely identified by his/her reader-id and we also record his/her name, phone number and address. The address is composed of street and suburb.
- A publisher is uniquely identified by its publisher-id. For each publisher, the name is also recorded.
- An author is uniquely identified by his/her author-id. For each author, the name, phone number and birth date are also recorded.
- A reader can borrow zero or more books and a book can be borrowed by zero or more readers. Thus, we need to record the starting date and ending date for the borrowing relationship.
- A publisher can publish zero or more books and a book is published by exactly one publisher. We also need to record the date of publication.
- An author can write zero or more books and a book is written by one or more authors.

Final Exercise: A Library Database

- A **book** is uniquely identified by its *book-id*. For each book, we also record its *title*, *price*, and *availability*.
- A **reader** is uniquely identified by his/her *reader-id* and we also record his/her *name*, *phone-number* and *address*. The address is composed of *street* and *suburb*.
- A **publisher** is uniquely identified by its *publisher-id*. For each publisher, the *name* is also recorded.
- An **author** is uniquely identified by his/her *author-id*. For each author, the *name*, *phone-number* and *birth-date* are also recorded.
- A **reader** can borrow zero or more books and a book can be borrowed by zero or more readers. Thus, we need to record the *starting-date* and *ending-date* for the borrowing relationship.
- A **publisher** can publish zero or more books and a book is published by exactly one publisher. We also need to record the *date-of-publication*.
- An **author** can write zero or more books and a book is written by one or more authors.

Exercise

Entity focused

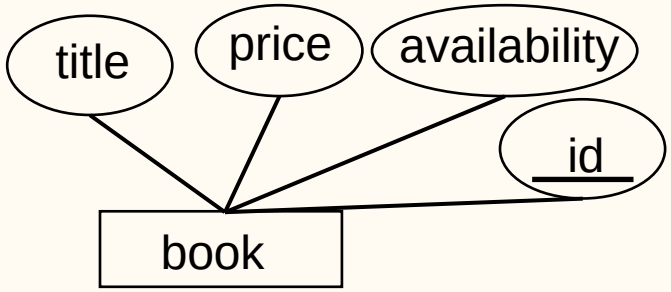
- A **book** is uniquely identified by its *book-id*. For each book, we also record its *title*, *price*, and *availability*.
- A **reader** is uniquely identified by his/her *reader-id* and we also record his/her *name*, *phone-number* and *address*. The address is composed of *street* and *suburb*.
- A **publisher** is uniquely identified by its *publisher-id*. For each publisher, the *name* is also recorded.
- An **author** is uniquely identified by his/her *author-id*. For each author, the *name*, *phone-number* and *birth-date* are also recorded.

Relation focused

- A **reader** can borrow zero or more books and a book can be borrowed by zero or more readers. Thus, we need to record the *starting-date* and *ending-date* for the borrowing relationship.
- A **publisher** can publish zero or more books and a book is published by exactly one publisher. We also need to record the *date-of-publication*.
- An **author** can write zero or more books and a book is written by one or more authors.

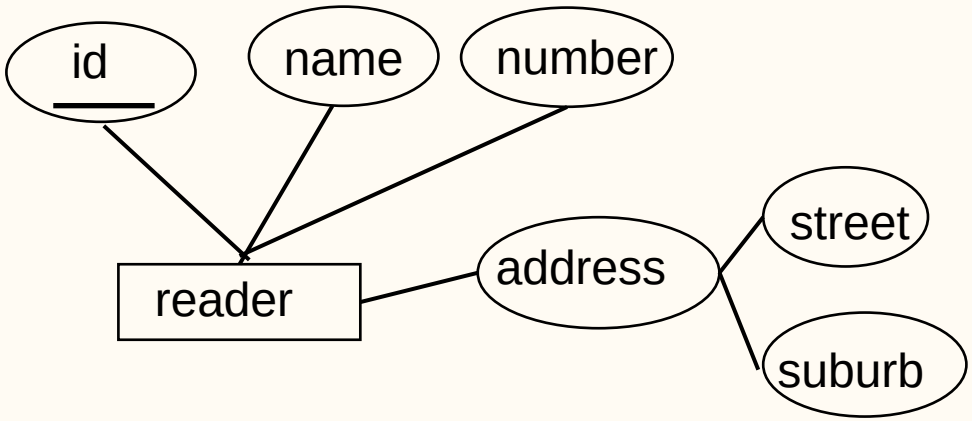
Exercise (1/7)

- A book is uniquely identified by its book id. For each book, we also record its title, price, and availability.



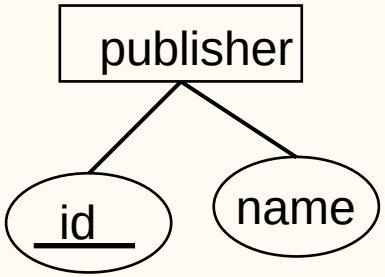
Exercise (2/7)

- A reader is uniquely identified by his/her reader id and we also record his/her name, phone number, dob and address. The address is composed of street and suburb.



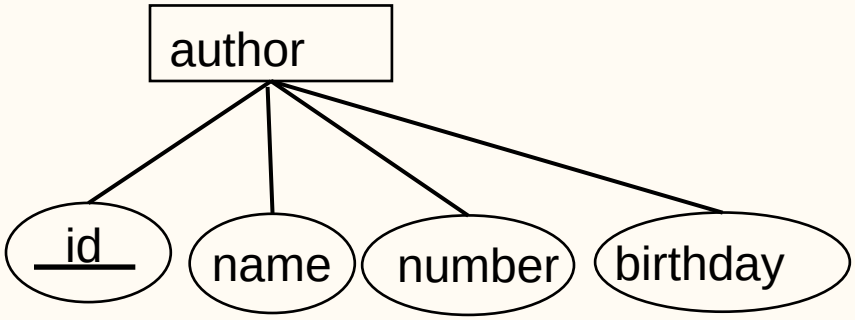
Exercise (3/7)

- A publisher is uniquely identified by its publisher id. For each publisher, the name is also recorded.



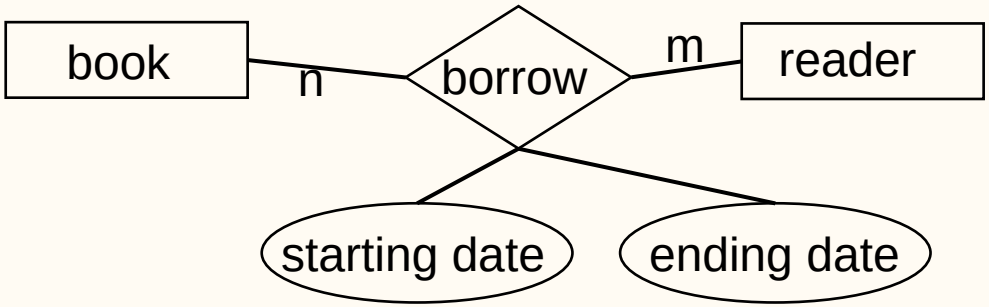
Exercise (4/7)

- An author is uniquely identified by his/her author id. For each author, the name, phone number and birth date are also recorded.



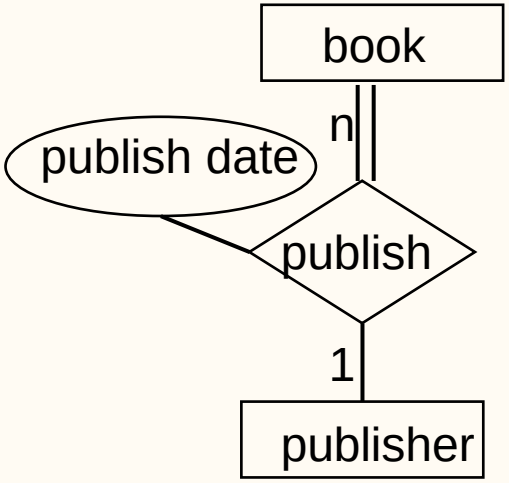
Exercise (5/7)

- A reader can borrow zero or more books and a book can be borrowed by zero or more readers. Thus, we need to record the starting date and ending date for the borrowing relationship.



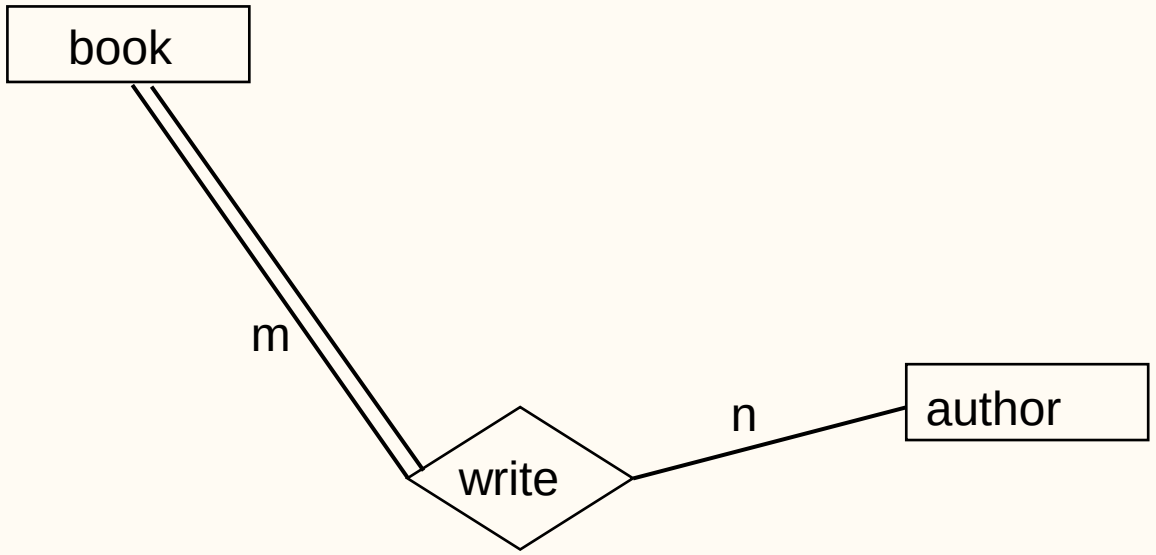
Exercise (6/7)

- A publisher can publish zero or more books and a book is published by exactly one publisher. We also need to record the date of publication.

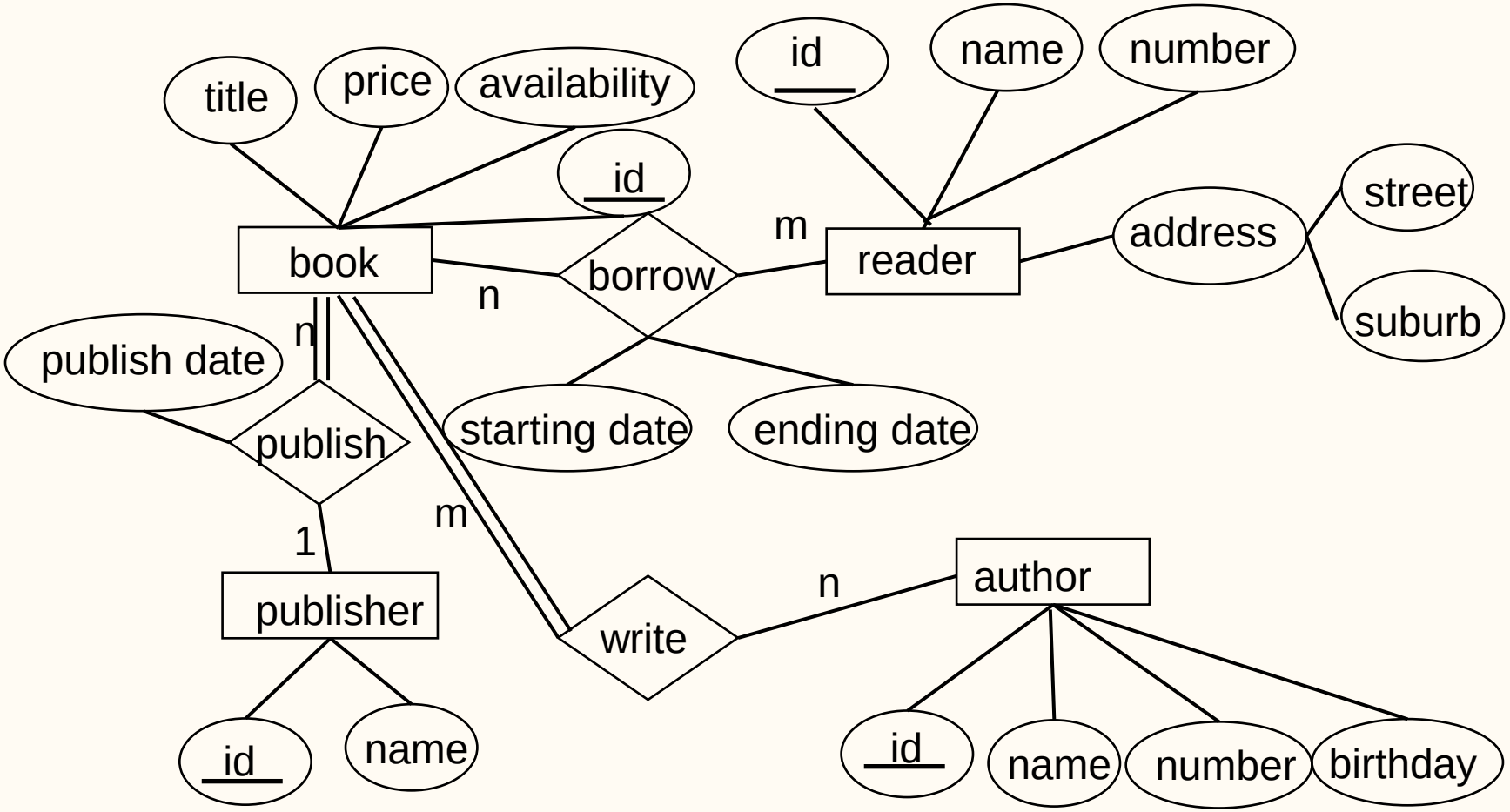


Exercise (7/7)

- An author can write zero or more books and a book is written by one or more authors.



Full Sample Solution



Summary

A high level process that converts applications into a data model

1. describe the information in a way that is suitable for database implementation (e.g., entities: students, courses, accounts, branches, patients, ...)
2. describe how information is related (e.g., John is enrolled in COMP3311, Andrew's account is held at Comm Bank)

Learning Outcome:

1. Given application requirements, how to ER-diagram to accurately reflect the application (and all the specific requirements).
2. Understand ER model as a solution for expressing any application requirement.

NOTE: You must use the notation in the lecture for this course, but not the one from the EER, and the weak entity example. Thanks.

End