COMP9120

Week 2: Conceptual Database Design

Semester 1, 2022

Dr Mohammad Polash
School of Computer Science

Contents developed by Dr Lijun Chang

THE UNIVERSITY OF SYDNEY

Acknowledgement of Country

I would like to acknowledge the Traditional Owners of Australia and recognise their continuing connection to land, water and culture. I am currently on the land of the Burramattagal people and pay my respects to their Elders, past, present and emerging.

I further acknowledge the Traditional Owners of the country on which you are on and pay respects to their Elders, past, present and future.

THE UNIVERSITY OF SYDNEY

Copyright warning

Outline

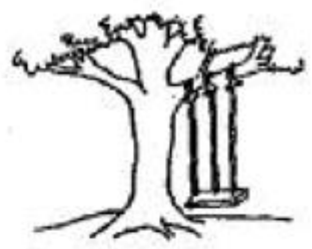
- Introduction to Conceptual Database Design

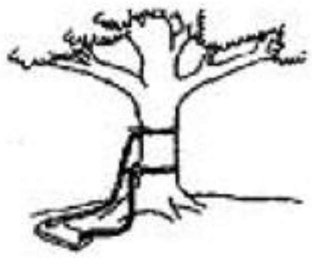
- Entity Relationship Model
  - Notation and usage
  - Entity and Relationship types, attributes
  - Key, participation and cardinality constraints
  - Weak entities, IsA hierarchies, aggregation

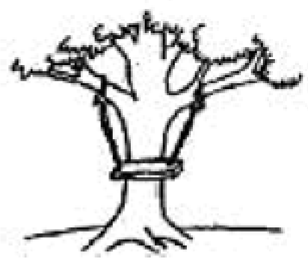
Conceptual Database Design

- Purpose of conceptual database design
  - Agree on the structure of database before deciding on a particular implementation

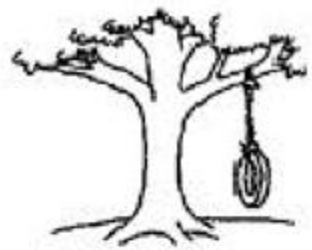
As proposed by the project sponsor.


As specified in the project request.


As designed by the senior architect.


As produced by the engineers.

As installed at the user's site.


What the customer really wanted.

Database Design Sequence

1. Requirements Analysis	Understand… - what data is to be stored - what applications must be built - what operations are most frequent
2. Conceptual Design	Develop… - high-level description of the data closely matching how users think of the data - Works as communication vehicle Today
3. Logical Design	Convert… - conceptual design into a logical database schema
4. Schema Refinement	Refine… - Identify problems in current schema & refine
5. Physical Design	Convert… - logical schema into a physical schema for a specific DBMS and tuned for app.
6. App & Security Design	Identify User who plays what Roles, in what Workflows, & secure … - What roles are played by different system entities in system processes, and what permissions should be given to these roles?

Conceptual Database Design

- Goal: Specification of database schema
- **Conceptual Database Design**: A technique for understanding and capturing business information requirements *graphically*
- It does *not* imply how data is implemented, created, modified, used, or deleted.
  - Works as communication vehicle between technical people and non-technical people
  - Facilitates planning, operation & maintenance of various data resources

Entity Relationship Model

- First designed by Peter Chen in 1976
  - Several variations have since appeared
  - Here: **enhanced** or **extended E-R model**

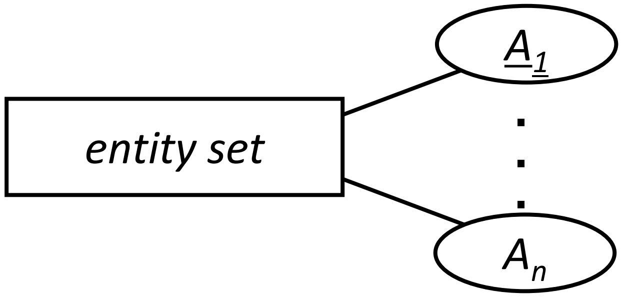
- A data modelling approach that depicts the associations among different categories of data within a business or information system.
  - What are the ***entities*** and ***relationships*** in the enterprise?
  - What information about these entities and relationships should we store in the database?
  - What are the *integrity constraints* or *business rules* that hold?

- A database ‘schema’ in the ER Model is represented pictorially (***ER diagrams***).
  - We can convert an ER diagram into a logical (e.g., relational) schema.

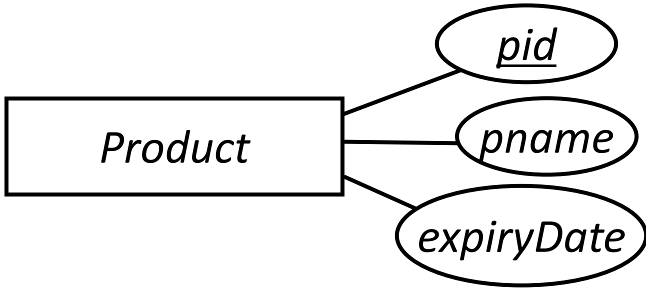
Entities & Entity set

- **Entity**: an individual object, e.g., a person, place, object, event, or concept about which you want to gather and store data.
  - it must be distinguishable from other entities
  - Example: John Doe, unit COMP9120, account 4711

- **Entity Type** (also: **entity set**): a collection of entities that share common properties or characteristics
  - Example: students, courses, accounts
  - Rectangle represents entity set



- **Attribute**: describes one aspect of an entity set
  - Descriptive properties possessed by all members of an entity set
  - Example: people have *IDs*, *names* and *addresses*
  - depicted by an ellipses



Entity Schema

- ***Domain***: possible values of an attribute
  - In contrast to relational model, values can be complex / set-oriented!
    - **Simple** and **composite** attributes.
    - **Single-valued** and **multi-valued** attributes

- ***Key***: *minimal* set of attributes that uniquely identifies an entity in the set (several such **Candidate Keys** possible)
  - Example: each student is uniquely identified by the student ID.
  - One chosen as **Primary Key** (PK) => depicted by <u>underlining</u> the attributes.

- ***Entity Schema***: entity set name, attributes (+domains), PK

Identify Keys

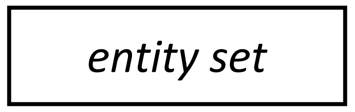
- Which one would you choose as a primary key?
  - Staff(staff_id, name, email, phone_number, address, dob)

- Which one would you choose as a primary key?
  - House(house_number, street_name, suburb, state, build in, price)

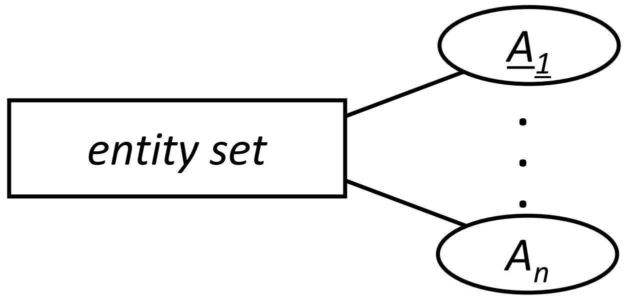
Graphical Representation in E-R Diagram

Symbols:

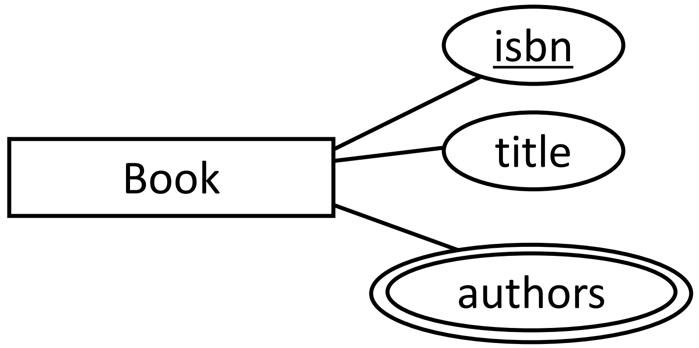
- Entity Sets represented by a rectangle

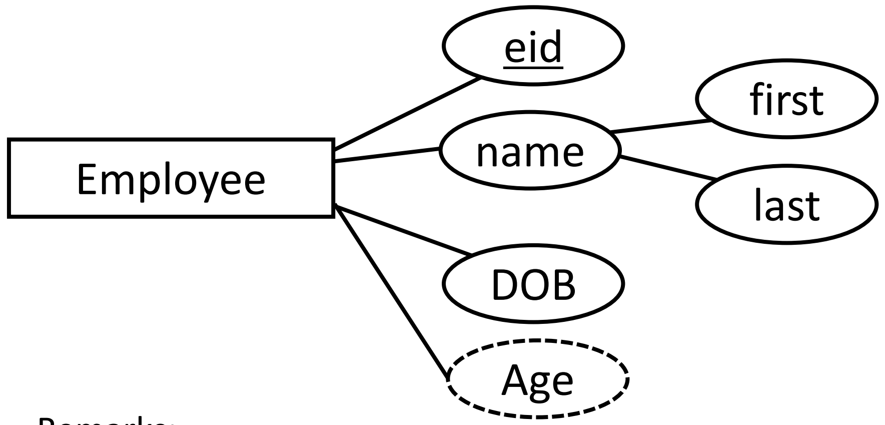


- Attributes depicted by ellipses
  - Keys are underlined
  - Double ellipses for multi-valued attributes



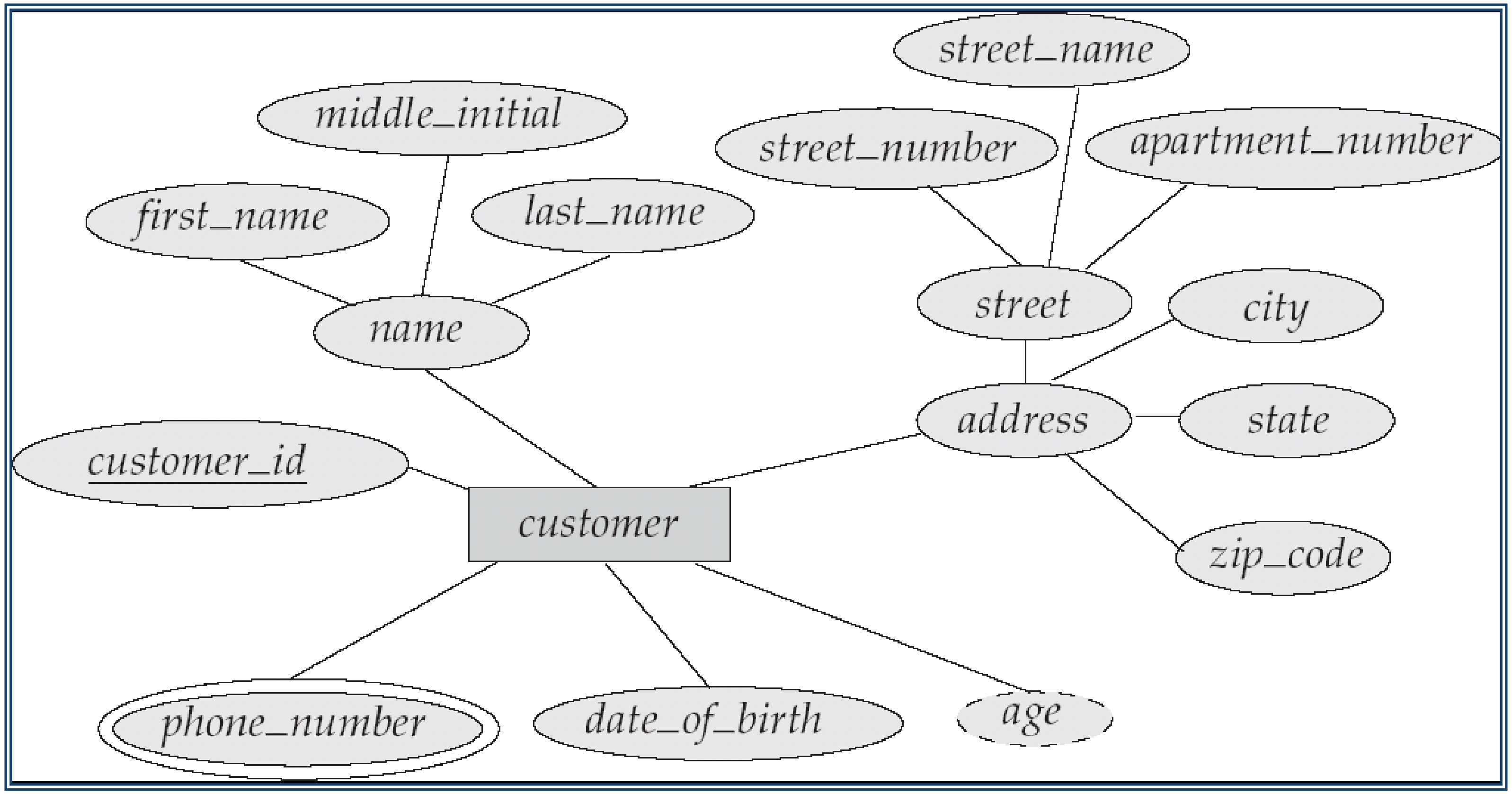
Examples:





Remarks:
Book.authors is a *multi-valued attribute*;
Employee.name is a *composite attribute*.
Employee.Age is a *derived attribute*

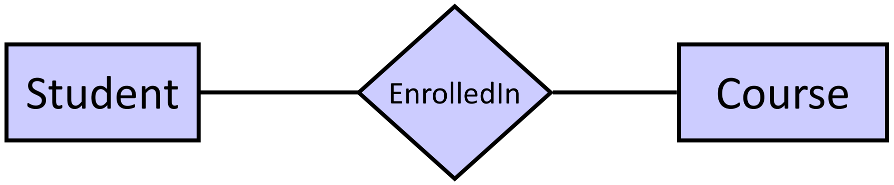
Entity Set


street_name
middle_initial
street_number
apartment_number
first_name
last_name
street
city
name
address
state
customer_id
customer
zip_code
phone_number
date_of_birth
age

Relationships

- **Relationship**: relates two or more entities
  - Example: John *is enrolled in* COMP9120

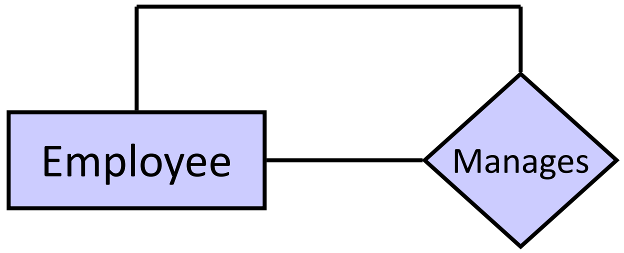
- **Relationship Type** (**Relationship Set**): set of similar relationships
  - Formally: a relation among $n \geq 2$ entities, each from entity sets:
    $\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$
  - Example: **Student** (entity set) related to Courses (entity set) by **EnrolledIn** (relationship type).
  - Diamond represents relationship type



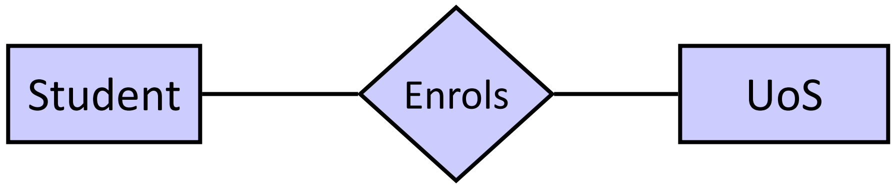
Relationship Degree

› Degree of a Relationship:
  # of entity types involved

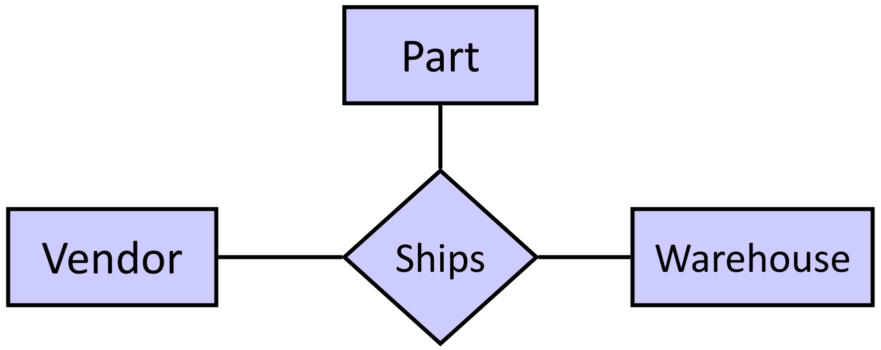
- Unary Relationship (Recursive)



- Binary Relationship



- Ternary Relationship

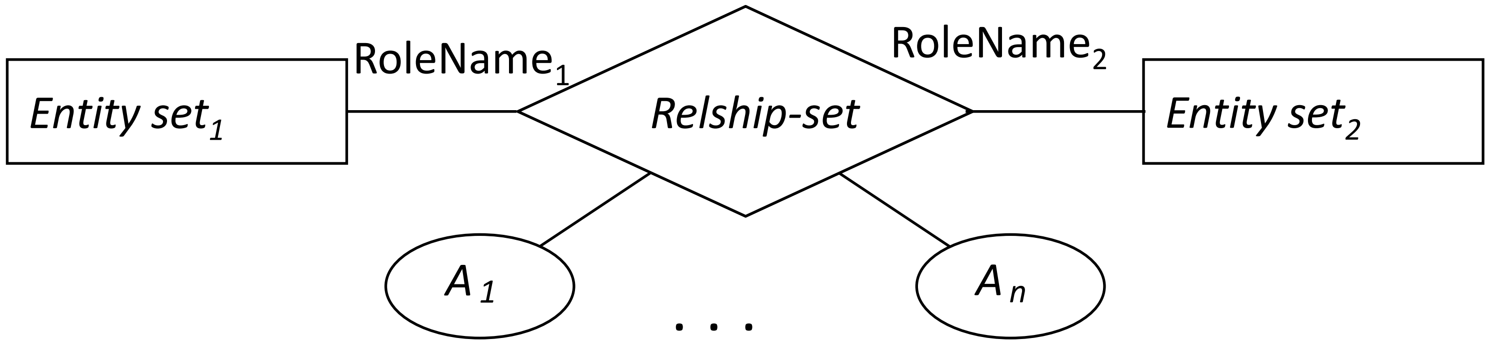


Relationship Attributes and Roles

- **Relationship-Attribute:** Relationships can also have additional properties
  - E.g., John enrols in COMP9120 *in* the first semester of 2021
  - John and COMP9120 are related
  - 2021sem1 describes the relationship - value of the *Semester* attribute of the **EnrolledIn** relationship set
- **Relationship-Role:** Each participating entity can be named with an explicit role
  - E.g. John is value of *Student* role, COMP9120 value of *Subject* role
  - useful for relationship that relate elements of the same entity type
  - Example: **Supervises(Employee:Manager, Employee)**
- **Relationship Type Schema**:
  - Relationship name
  - Role names (or names of participating entity sets) – this is optional
  - Relationship attributes (+domains)

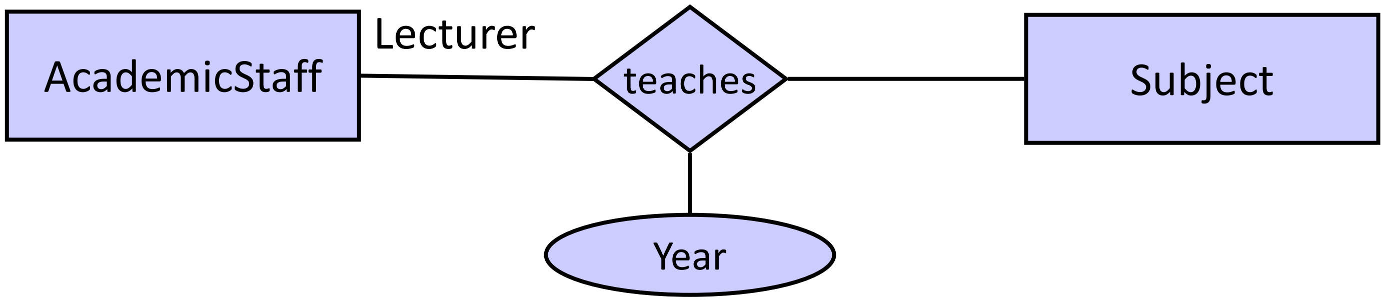
Graphical Representation of Relationships in ERD

Symbol:



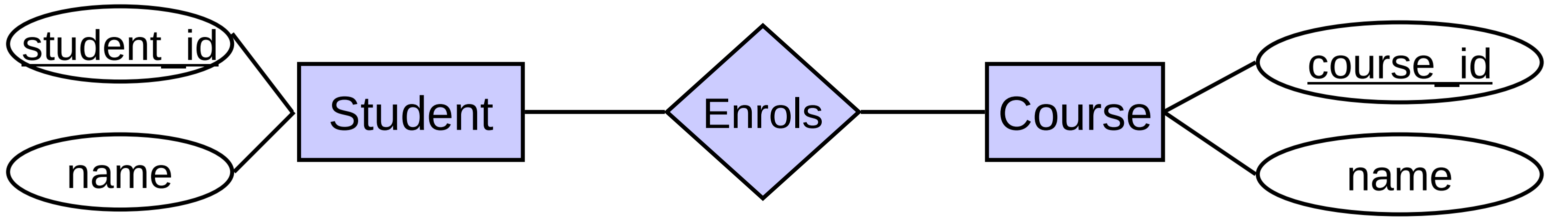
- Diamonds represent relationship set
- Lines link attributes to entity sets and entity sets to relationship set.
- Roles are edges labeled with role names

Example



Key of Relationship Set

› The combination of the primary keys of the participating entity sets forms a **super key** (superset of a key) of a relationship.
  - There can only be ***one relationship for every unique combination of entities***
  - Example: (student_Id, course_id) is the super key of *Enrols*

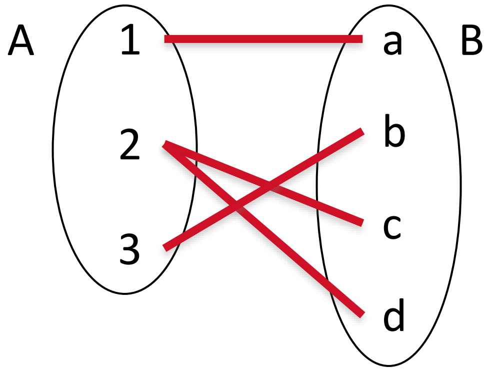


  - One must consider the ***mapping cardinality of the relationship*** when deciding what are the candidate keys
    - Consider **Works_In**: An employee can work in many departments; a department can have many employees.
    - In contrast, each department has at most one manager
  - We do not represent keys of relationship set in E-R diagram

Mathematical Definition of Relationship?

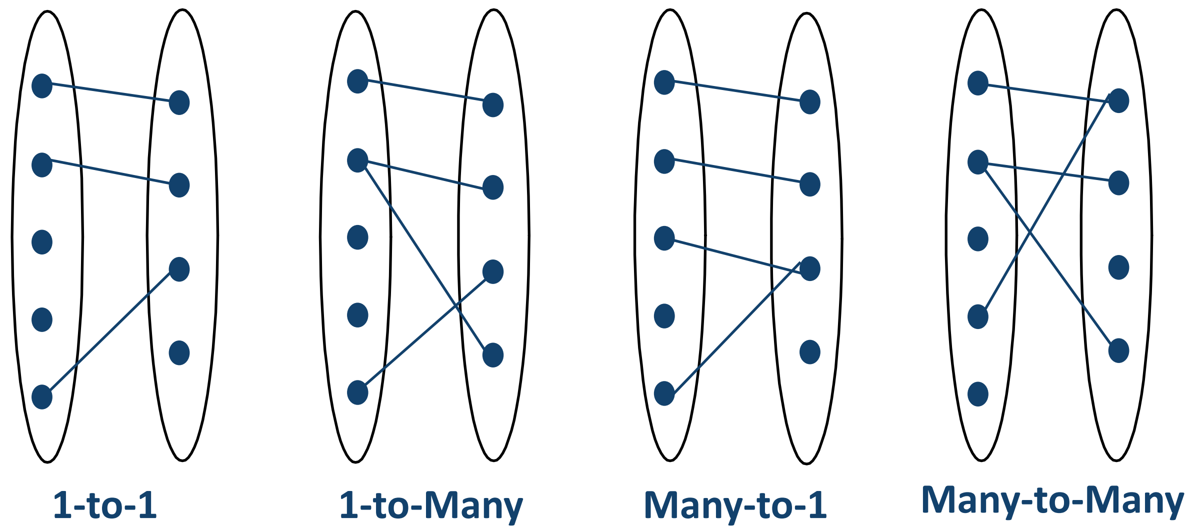
› In order to see the mapping cardinality of the relationship, let's consider the mathematical definition of relationship

- Let A, B be sets
  - *A={1,2,3},   B={a,b,c,d},*
- A x B (the ***cross-product***) is the set of all pairs (x,y)
  - *$A \times B$ = {(1,a), (1,b), (1,c), (1,d), (2,a), (2,b), (2,c), (2,d), (3,a), (3,b), (3,c), (3,d)}*
- **A <u>relationship</u> is a subset of A x B**
  - *R = {(1,a), (2,c), (2,d), (3,b)}*



Mapping Cardinality of Relationships

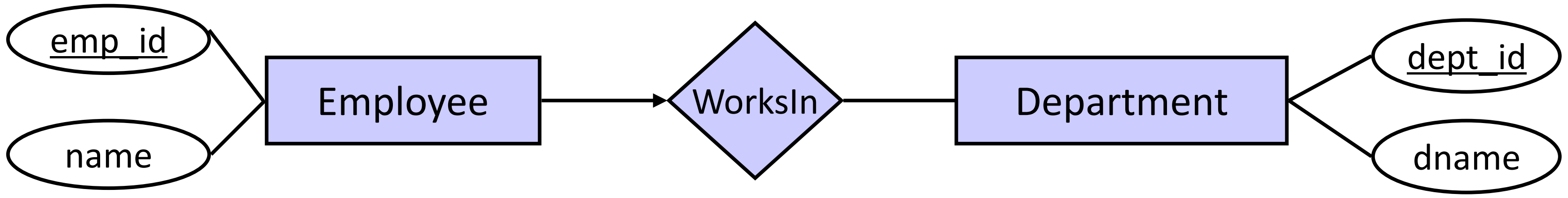
- We find examples of each style of relationship
  - Think carefully about both directions:
  - To how many instances of B can a given instance of A be related?
  - To how many instances of A can a given instance of B be related?
- Warning: natural language can be confusing
  - Many-to-1 means each of A is related to *at most* 1 of B
- *1 means each element of the* ***opposite*** *set appears at most once in the relationship*



Multiplicities are depicted in E-R diagrams as constraints…

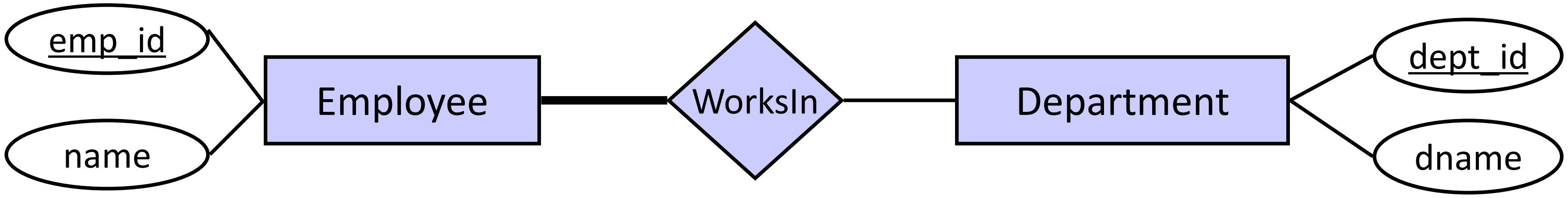
Key Constraints

- If, for a particular entity set, each entity participates in *at most one* instance of a relationship, the corresponding role is a key of relationship type
  - E.g., *Employee* role is unique in *WorksIn*
    - called: **many-to-one** or **N:1 relationship**
- Representation in E-R diagram: arrow
- Example: An employee works in at most one department.



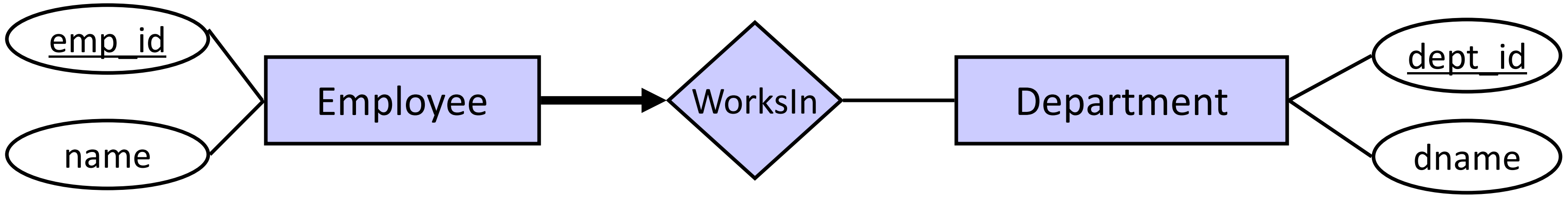
Participation Constraint

- If every entity participates in *at least one* instance of a relationship, a *participation constraint* holds:
  - also called a **total participation** of *E* in *R*
    - A participation that is not total is said to be **partial**
- Representation in E-R diagram: thick line
- Example: Every employee works in at least one department



Participation and Key Constraint

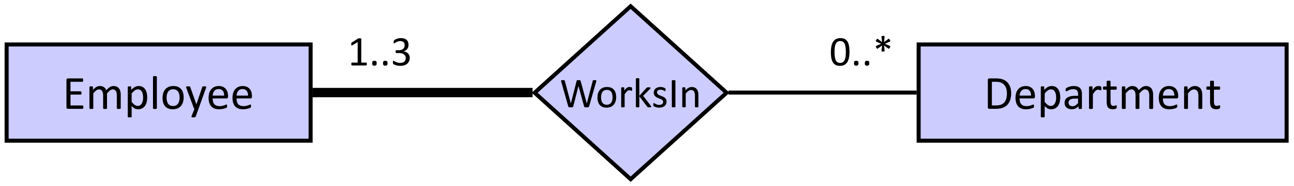
- If every entity participates in <u>*exactly one*</u> relationship, both a participation and a key constraint hold.
- Representation in E-R diagrams: thick arrow
- Example: Every employee works in exactly one department
  - Again: N:1 relationship



THE UNIVERSITY OF SYDNEY

Cardinality Constraints

- Generalisation of key and participation constraints
- A **cardinality constraint** for the participation of an entity set E in a relationship R specifies how often an entity of set E participates in R at least (minimum cardinality) and at most (maximum cardinality).
  - In an ER-diagram, we annotate the edge between an entity set E and relationship R with min..max, where min is the minimum cardinality and max the maximum cardinality. If no maximal cardinality is specified, we set ‘*’ as max number ("don’t care").
- Example: Every employee works in 1 to 3 departments.



Let’s take a break!

When you come back, please complete the poll

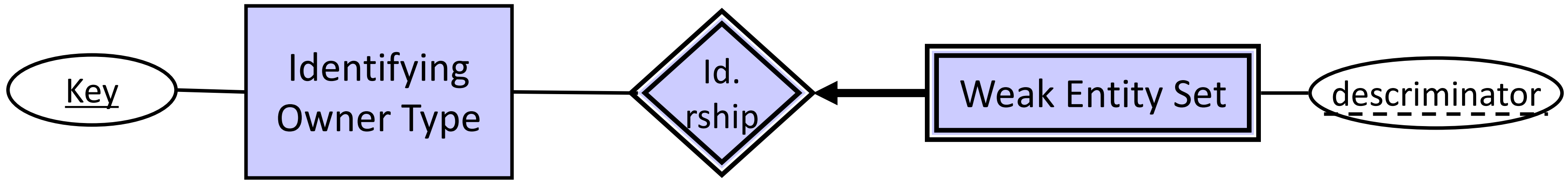
Weak Entities

- ***Weak entity type***: An entity type that does not have a self-contained primary key.
  - Can be seen as an **exclusive 'part-of' relationship**
  - Its existence depends on the existence of an ***identifying owner*** entity
  - The weak entity ***must***:
    - relate to the identifying owner entity set via a one-to-many *identifying relationship type* from the identifying owner entity set to the weak entity set
    - have total participation in the identifying relationship type
  - Example:

    payment of a loan
- The ***discriminator*** (or ***partial key***) of a weak entity type is the set of attributes that distinguishes among all the entities of a weak entity type related to the same owning entity.
- The primary key of a weak entity type is formed by the primary key of the strong entity type(s) on which the weak entity type is existence dependent, plus the weak entity type's discriminator.

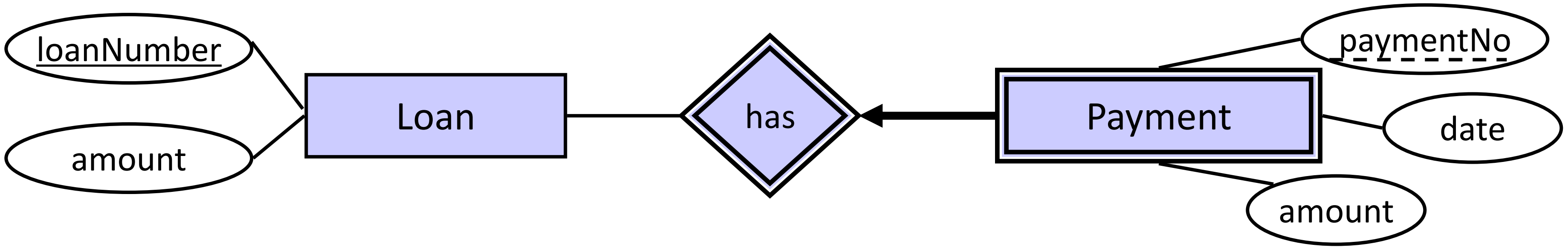
Representation of Weak Entity Types

Symbols:



- We depict a weak entity set by double rectangles.
- Identifying relationship depicted using a double diamond
- underline the discriminator of a weak entity set with a dashed line

Example:



- paymentNumber: discriminator of the payment entity set
- Primary key for payment: (loanNumber, paymentNumber)

Enhanced E-R Model

- ER model in its original form did not support
  - SPECIALIZATION/ GENERALIZATION
  - ABSTRACTIONS (‘aggregation’)

- This leads to the development of ‘Enhanced’ ER model
  - Includes all modelling concepts of basic ER
  - Additional some object-oriented concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
  - The resulting model is called the enhanced-ER or Extended ER (E2R or EER) model
    - used to model applications more completely and accurately if needed

- When we talk about E-R model, we always mean EER model

Generalisation/ Specialisation

- Arranging of entity sets in a type hierarchy.
  - Determine entity sets whose set of properties are a subset of another entity set

- Definition **Generalisation** / **Specialisation** / **Inheritance**:
  Two entity types *E* and *F* are in an IsA-relationship ("*F* is a *E*"), if

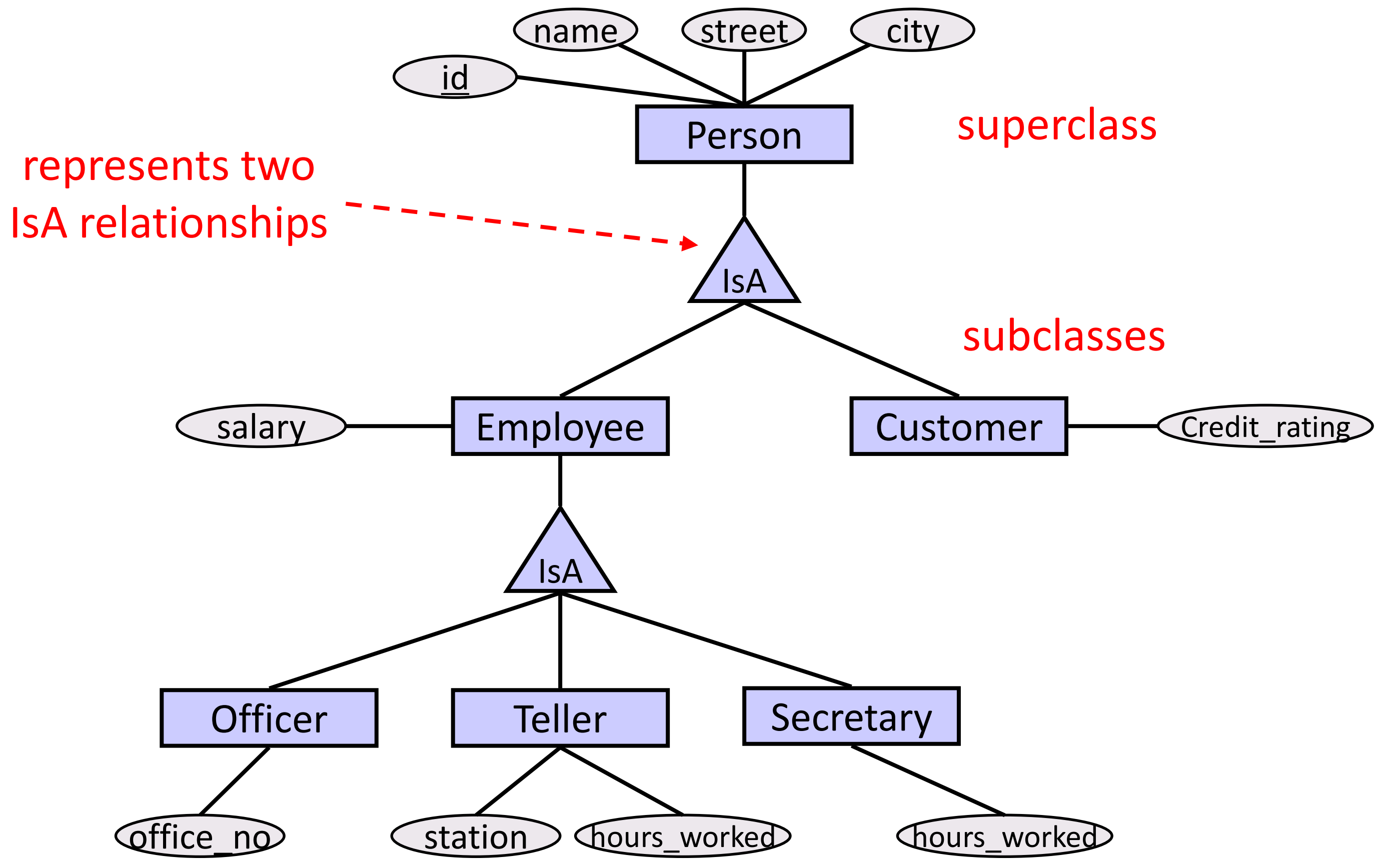
  (1) the set of attributes of *F* is a superset of the set of attributes of *E*, and

  (2) the entity set *F* is a subset of the entity set of *E* ("each *f* is an *e*")

- One says that *F* is a *specialisation* of *E* (*F* is **subclass**) and *E* is a *generalisation* of *F* (*E* is **superclass**).
  - Example: **Student** is a subclass of **Person**
- **Attribute inheritance** – a lower-level (subclass) entity type inherits all the attributes and relationship participations of its supertype.

- Depicted by a triangle component labelled IsA

Superclass / Subclass Example



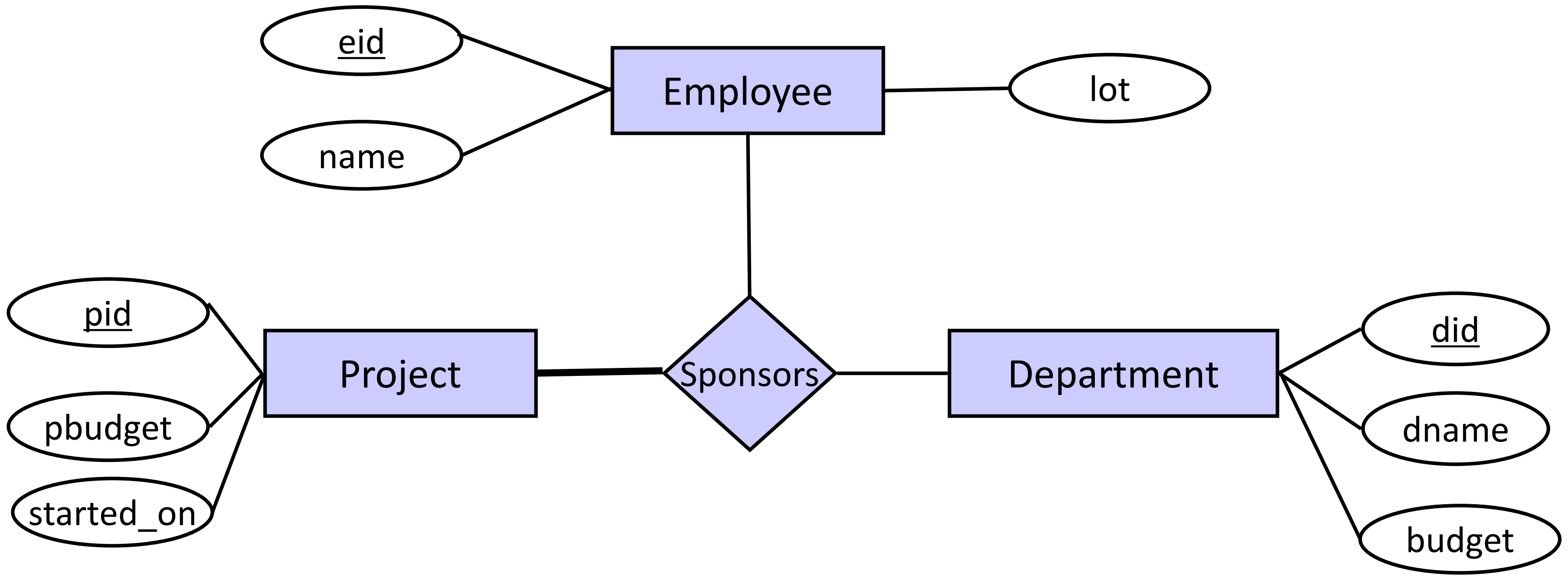
- Specialisation stems from a single entity set. It emphasizes differences among entities within the set by creating distinct lower-level entities
- Generalization proceeds from the recognition that a number of entity sets share some common features

Constraints on IsA Hierarchies

- We can specify *overlap* and *covering* constraints for ISA hierarchies:
- ***Overlap Constraints***
  - **Disjoint**
    - an entity can belong to only one lower-level entity set
    - Noted in E-R diagram by writing *disjoint* next to the IsA triangle
  - **Overlapping** (the default - *opposite to Ramakrishnan/Gehrke book*)
    - an entity can belong to more than one lower-level entity set
- ***Covering Constraints***
  - **Total**
    - an entity must belong to one of the lower-level entity sets
    - Denoted with a thick line between the IsA-triangle and the superclass
  - **Partial** (the default)
    - an entity need not belong to one of the lower-level entity sets

Aggregation

- Consider a ternary relationship *Sponsors*



  - Each project can be sponsored by one or more departments
  - A department can appoint one or more employees to monitor a sponsorship
  - Problems?

(Example adapted from Ramakrishnan & Gherke, DBMS, 3rd edition)

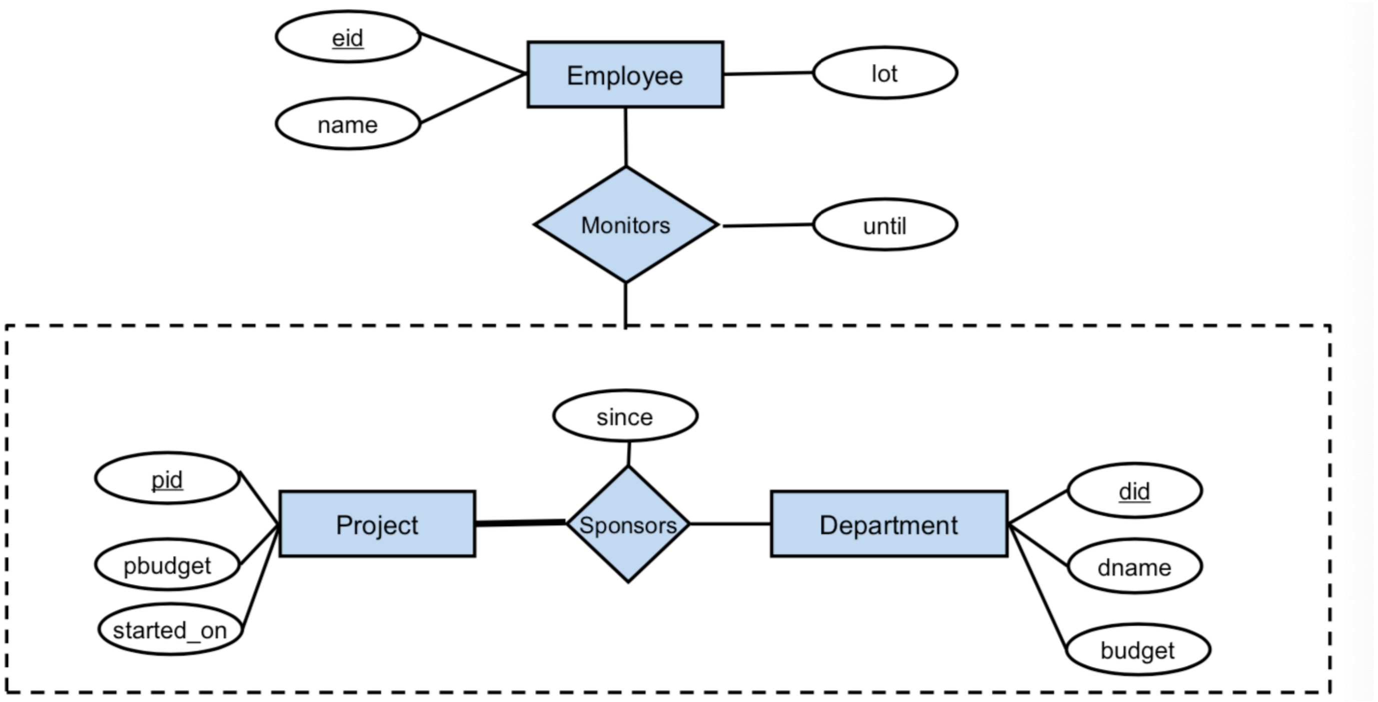
Aggregation (Cont'd)

- Relationship sets *Sponsors* really tries to model two relationships
  - It tries to model the fact that departments *sponsor* projects, but also that sponsorships can be *monitored.*
  - What if we want to add different attributes for each of these relationships?
    - It is more meaningful, and our model can communicate more information if we add such required attributes on the correct relationship.

- We can convey relationships between an entity set and another relationship set via *aggregation*
  - Treat relationship as an abstract entity – surround this with a dashed box (see next slide)
  - Allows relationships between relationship sets

E-R Diagram With Aggregation

- To convey a relationship between the *sponsors* relationship type and an employee entity type, we introduce aggregation via a *Monitors* relationship type.
  - Allows us to model that *sponsorships* start at a given time and that employees are assigned to *monitor* a *sponsorship* until a given time.



(Example from Ramakrishnan & Gherke, DBMS, 3rd edition)

This Week You Have Learned…

› **Conceptual database design using the E-R Model**

- Understanding and experience with conceptual database design using the entity-relationship model:
  - Basic Constructs: Entity, Attributes (single, composite, multivalued, derived), Relationships, Cardinality Constraints
  - Advanced Concepts: Weak Entities, Inheritance, Aggregation

IMPORTANT

- Many variations of ER diagrams are in use, and there is no widely accepted standard.

- Which notation should I use for labs/assessments?
  - Use the notation just outlined in these slides

THE UNIVERSITY OF SYDNEY

References

- Ramakrishnan/Gehrke (3rd edition)
  - **Chapter 2**
- Kifer/Bernstein/Lewis (2nd edition)
  - Chapter 4
- Ullman/Widom (3rd edition)
  - Chapter 4
- Silberschatz/Korth/Sudarshan (5th edition - ‘sailing boat’)
  - Chapter 6
- Elmasri/Navathe (5th edition)
  - Chapters 3 and 4

Next Week: Relational Model

- Readings:
  - **Ramakrishnan/Gehrke, Chapter 3 plus Chapter §1.5**
  - Kifer/Bernstein/Lewis, Chapter 3
  - Ullman/Widom, Chapter 2.1 - 2.3, Section 7.1 and Chapter 8.1-8.2

See you next week!

Remember to form groups