Entity Relationship Models 2

14 November 2011
Lecture 3

Topics for Today

- Entity Relationship Diagrams
  - Additional Features
- Conceptual Design using Entity Relationship Diagrams
  - for Large Enterprises
- Introduction to Relational Model

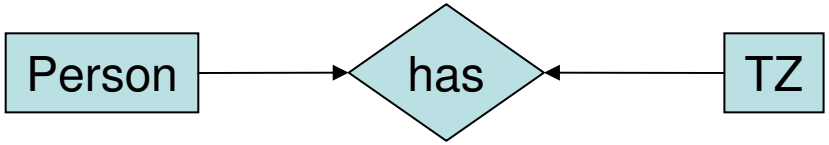
- Source: Ramakrishnan and Gehrke 2.4-2.7, 3.1

Additional Features of ERD

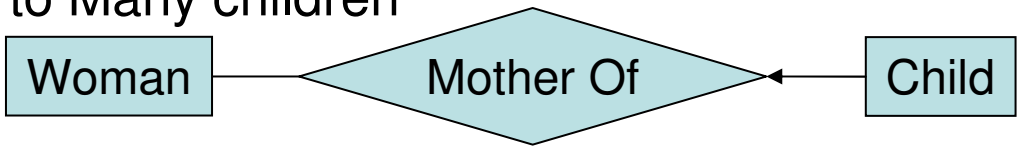
- ERDs can also represent more complex aspects of relationships:
  - Key Constraints
  - Participation Constraints
  - Weak Entities
  - Inheritance
  - Aggregation

Key Constraints

- Types of Key Constraints:
  - One to one
  - One to Many
  - Many to Many
- Examples:
  - One to One – 1 TZ to 1 person

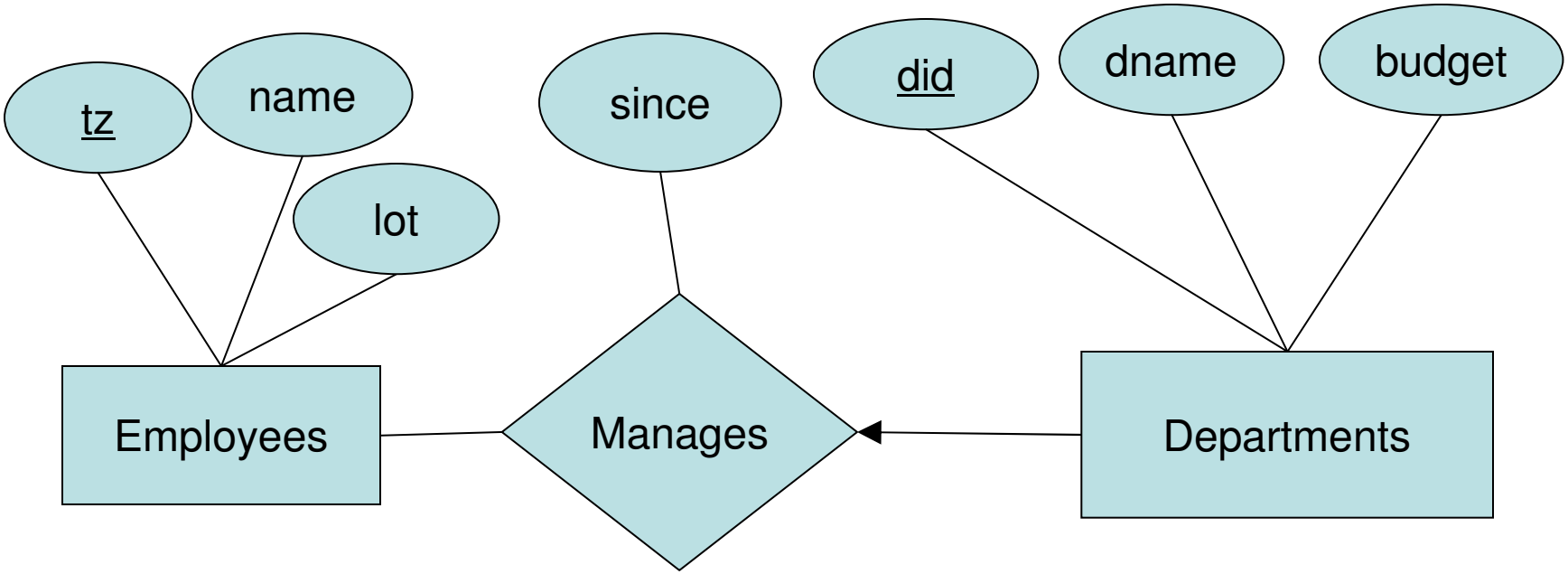


  - One to Many – 1 mother to Many children



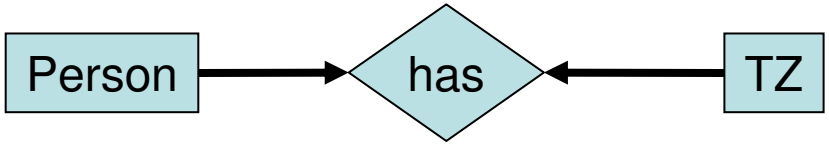
  - Many to Many – Many addresses to Many residents

Key Constraint Example

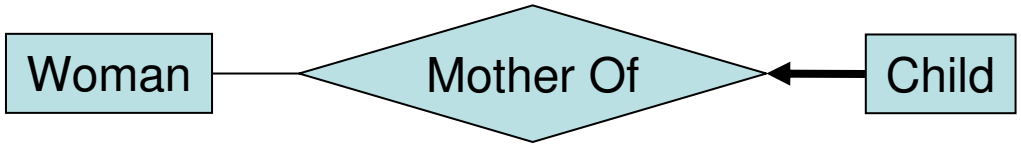


Participation Constraints

- Whether each member of an entity set must appear in the relationship
  - Thick line on its side if yes
- Examples:
  - Each person must have a TZ, every TZ is for a person

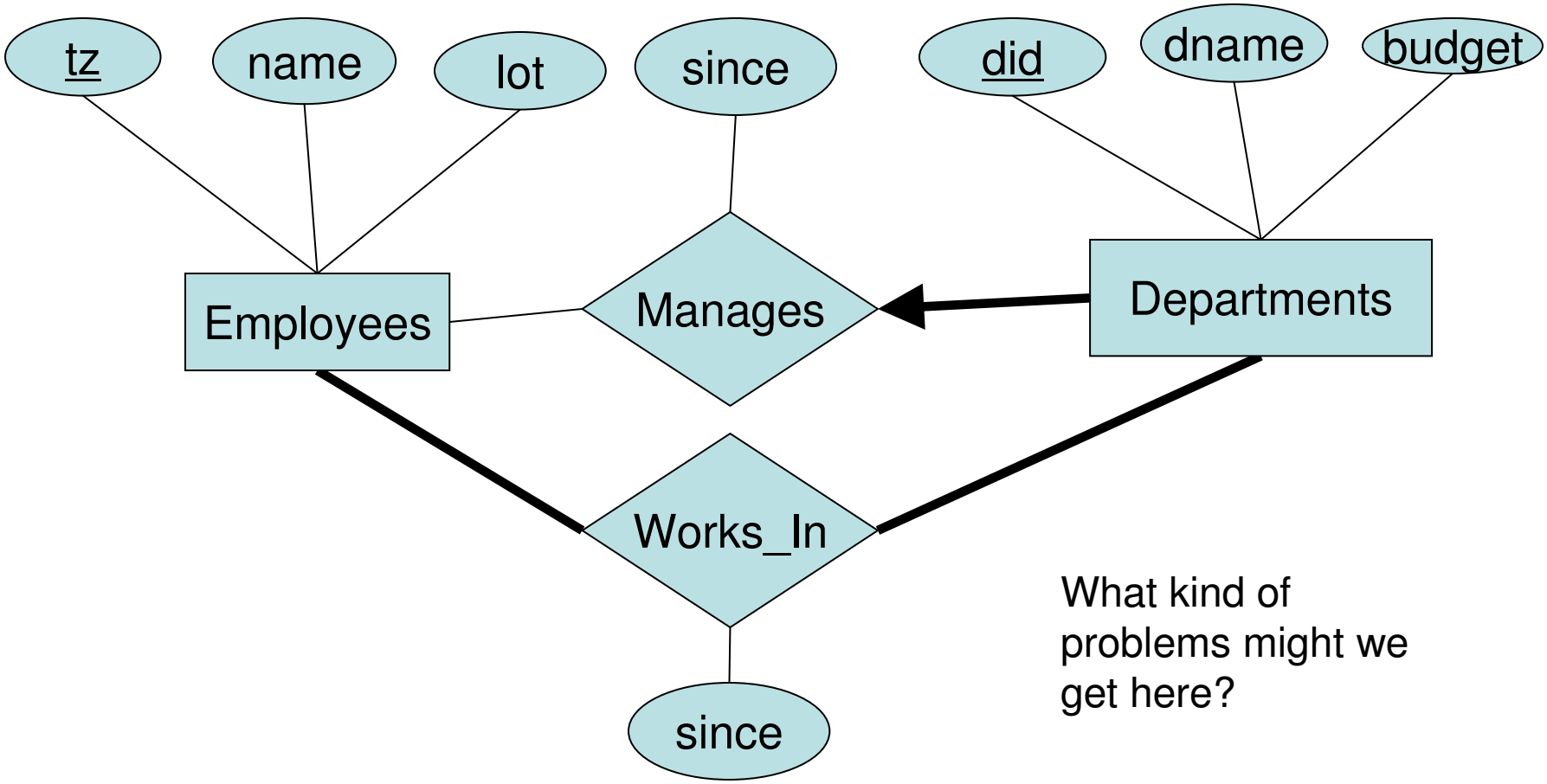


  - Each child must have a mother



  - Not everyone has an address, every address has someone there

Participation Example



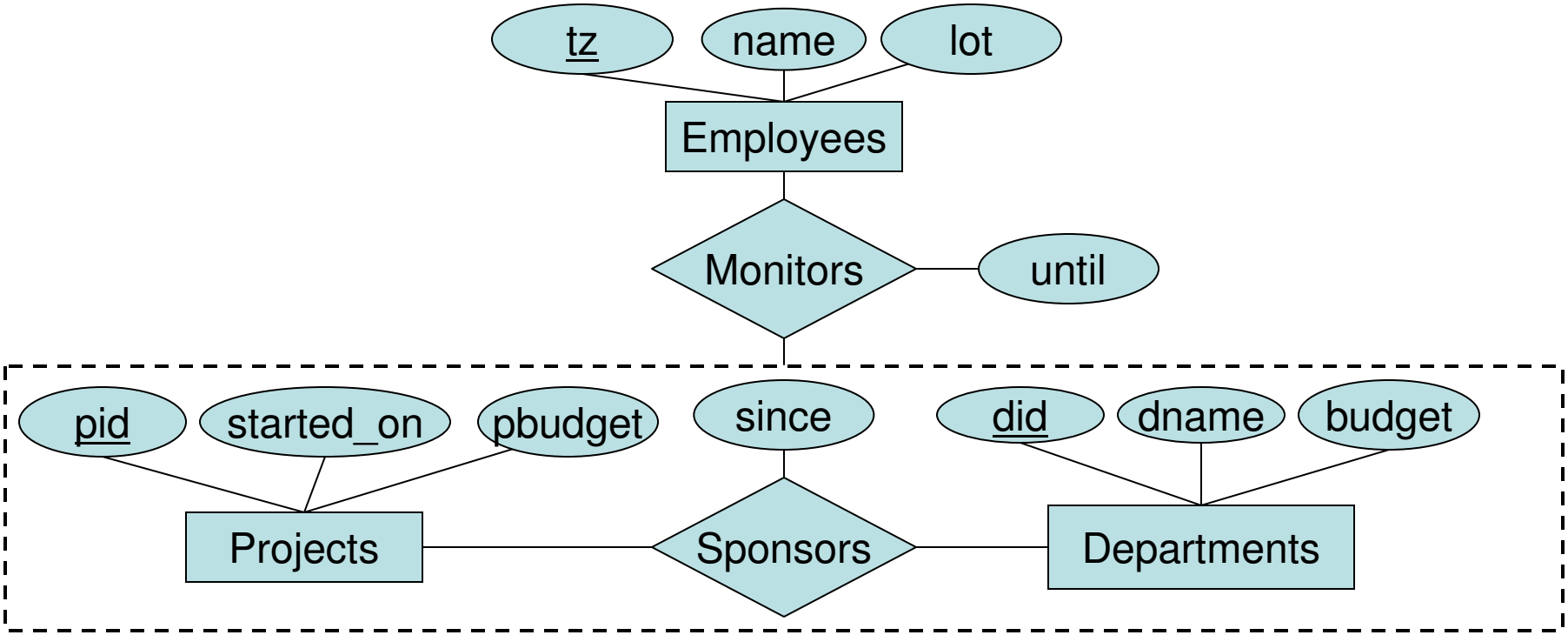
What kind of problems might we get here?

Aggregation

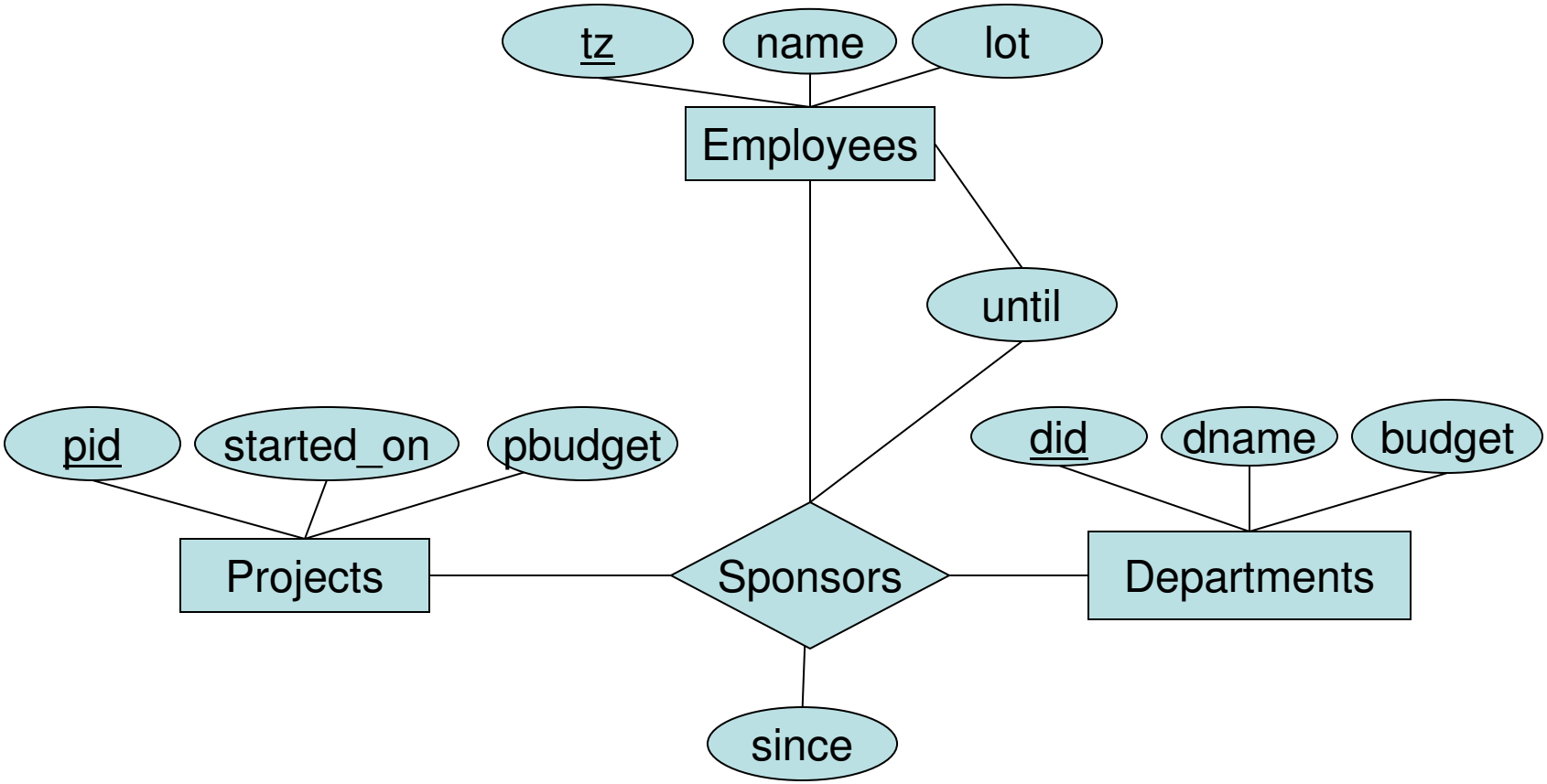
- Some complex relationships require relationships between relationships
  - Normally this is illegal in an ERD
  - Relationships can only connect entities
- We can allow such relationships by turning each instance of a relationship set into an entity
  - This is called **Aggregation**
  - We can then connect the aggregated relationship to other relationships
- BTW – it's hard to find simple examples of aggregation…
  - Think hard, because it might be doable with high order relationships instead

Aggregation Example

- Departments sponsor projects.
- There is an employee in charge of monitoring the project and its sponsorship until some date.



What else could we have done?



We need the aggregation to make the *until* attribute associated with the monitoring relationship alone.

So far

- Entity Relationship Diagrams
  - Additional Features
- Conceptual Design using Entity Relationship Diagrams
  - for Large Enterprises
- Introduction to Relational Model

Conceptual Design using ERD

- Making good design using ERDs requires considering some tradeoffs:
  - Is it an **entity** or an **attribute**?
  - Is it an **entity** or a **relationship**?
  - Should the relationship be **binary** or **ternary**?
  - Should we use **aggregation** or a **higher order relationships**?

- Let's discuss some guidelines

Entity vs. Attribute

1. If an attribute must be multivalued (with unknown cardinality) then it must be an entity

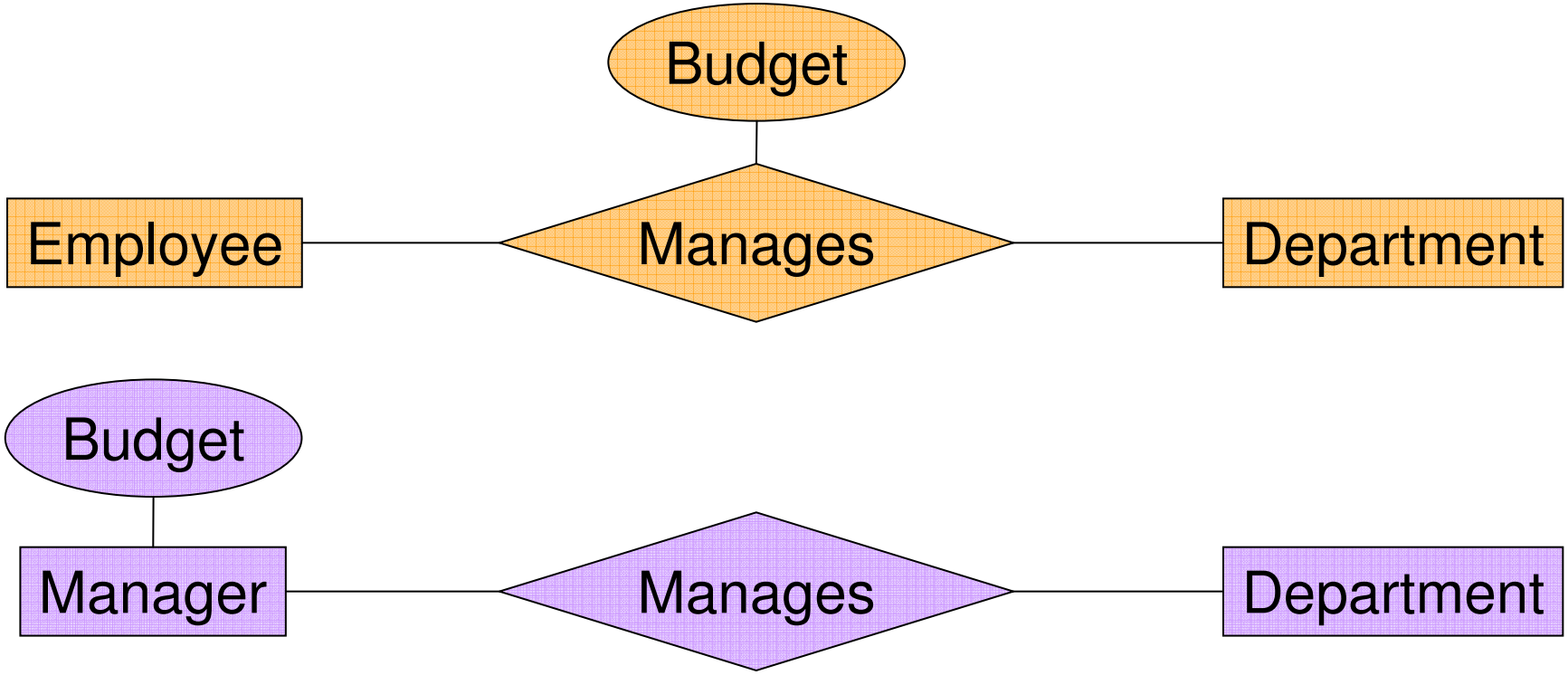
- **Example**: An Employee may have 1 or more addresses
  - Address must be an entity
  - If we know it's only 1-2, we can just make two Address attributes

2. If an attribute has structure or its own attributes, it must be an entity

- **Example**: An Address must maintain a start date and an end date to indicate its validity period
  - Address must be an entity since the end date is not an attribute of the Employee
  - If it's multivalued too, the first reason applies anyway

Entity vs. Relationship

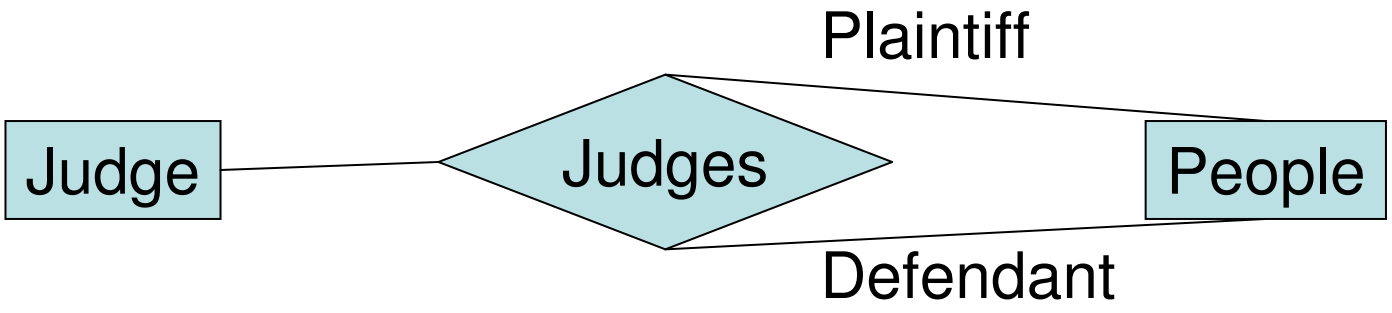
- If a relationship needs attributes which are shared between relationship instances, it must be an entity
- **Example:** A manager has a budget shared between multiple departments that she manages



Binary vs. Ternary

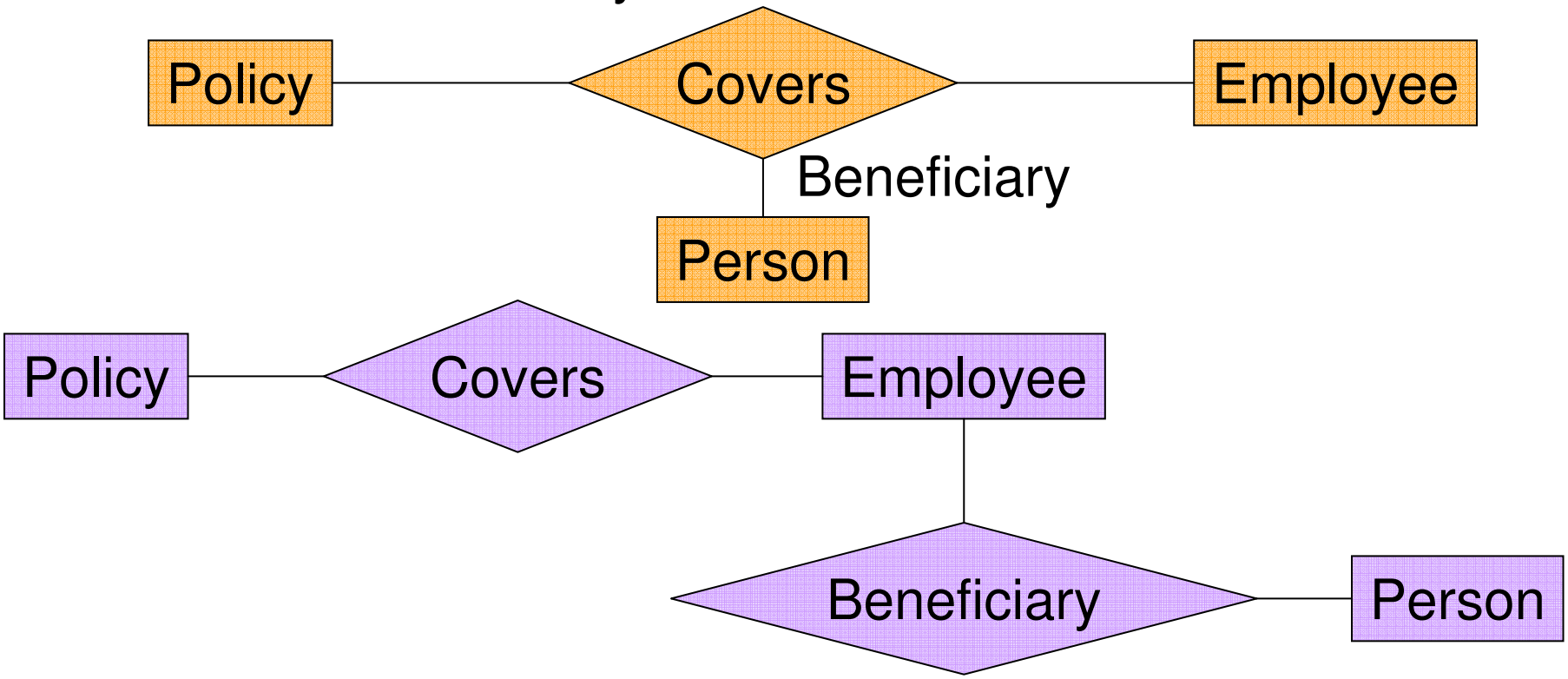
- If a relationship is between three entities, a ternary (or higher) relationship is required

- Example: The relationship between a Judge and two People



Binary vs. Ternary

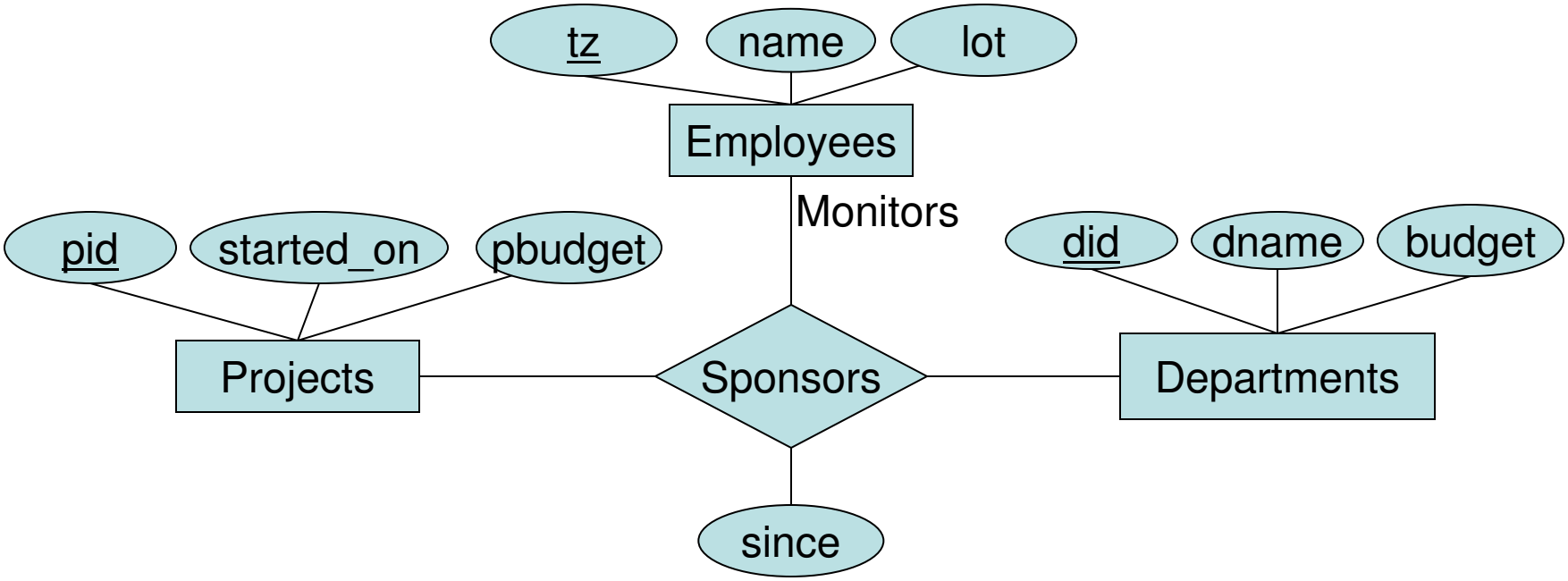
- If relationship is really a series of binary relationships, then just show them
- **Example:** A life insurance policy covers an individual and has a main beneficiary



Aggregation vs. Ternary

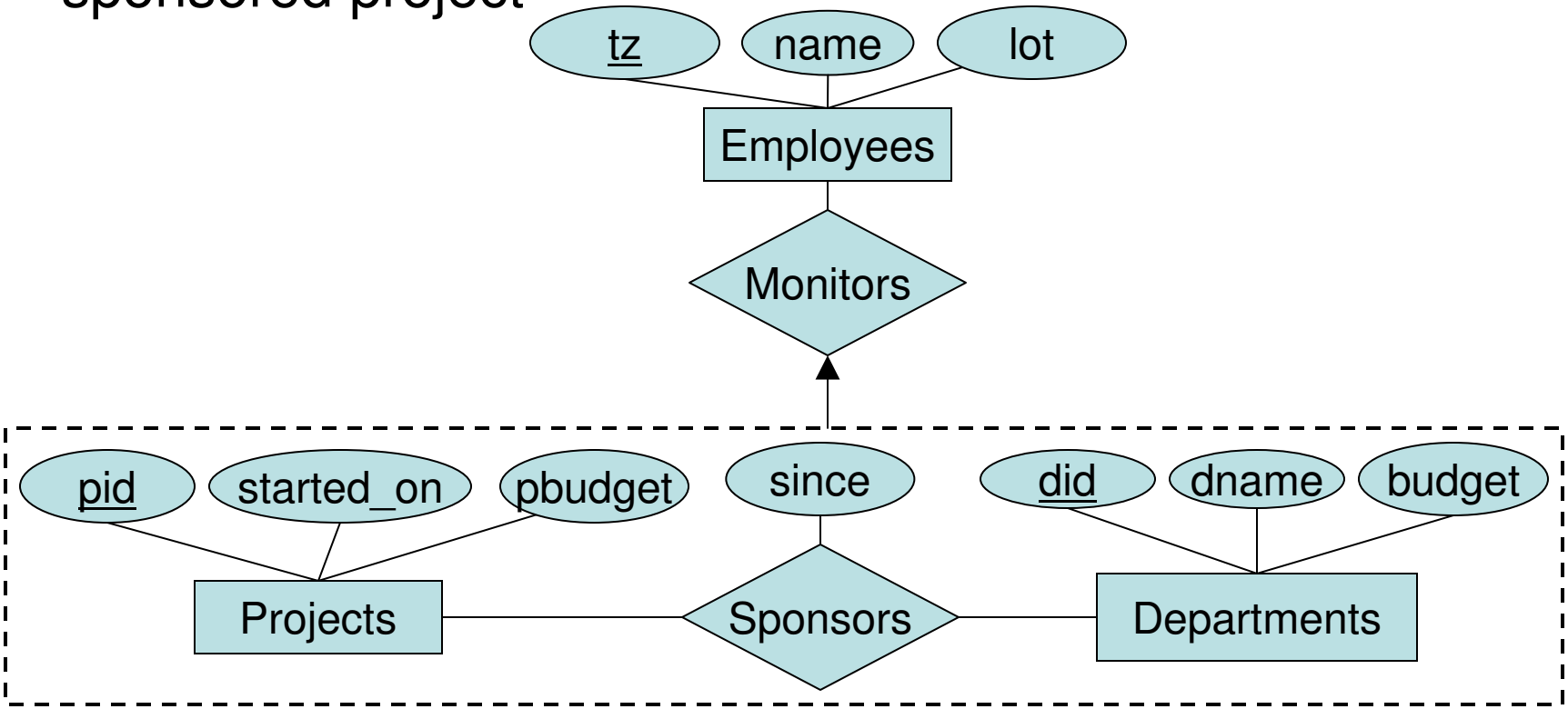
- If the relationship doesn't require its own attributes, we can use ternary (or higher order) instead of aggregation

- **Example:** If we didn't need the until attribute above, we could have made the relationship ternary instead



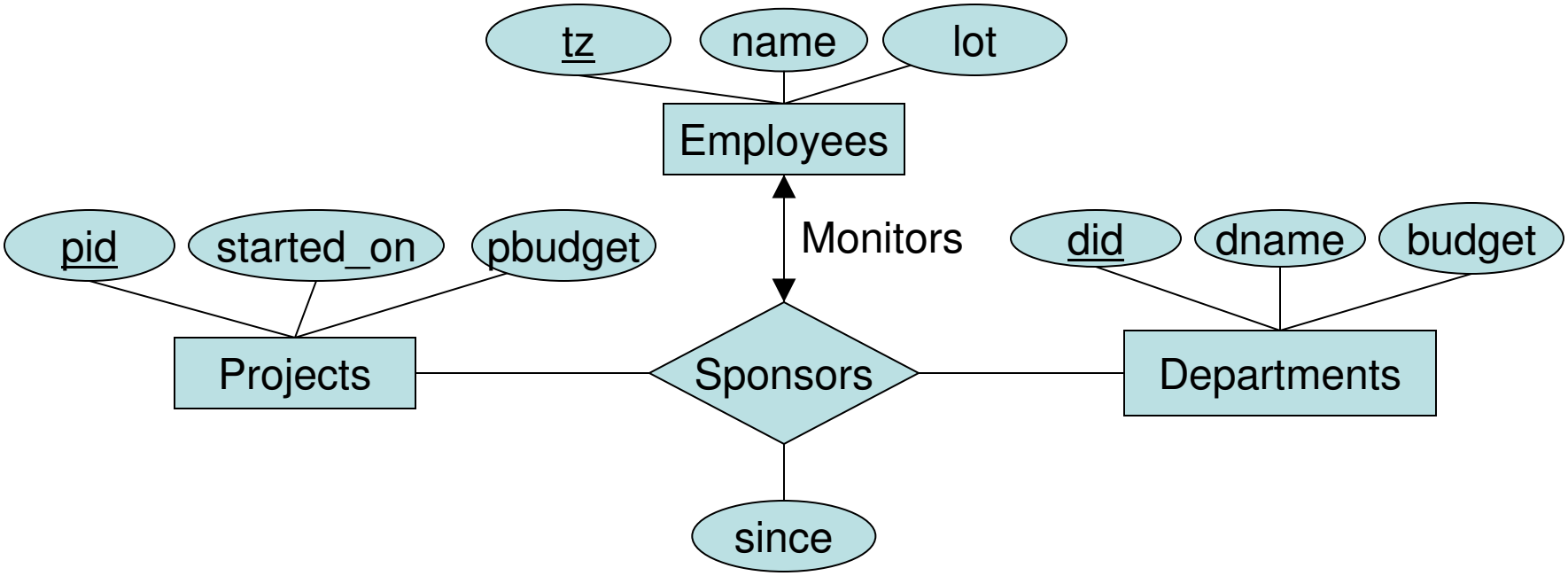
Aggregation vs. Ternary

- Some key constraints require aggregation
- **Example:** If at most one employee can monitor each sponsored project



Aggregation vs. Ternary

- Some key constraints require aggregation
- **Example:** If at most one employee can monitor each sponsored project

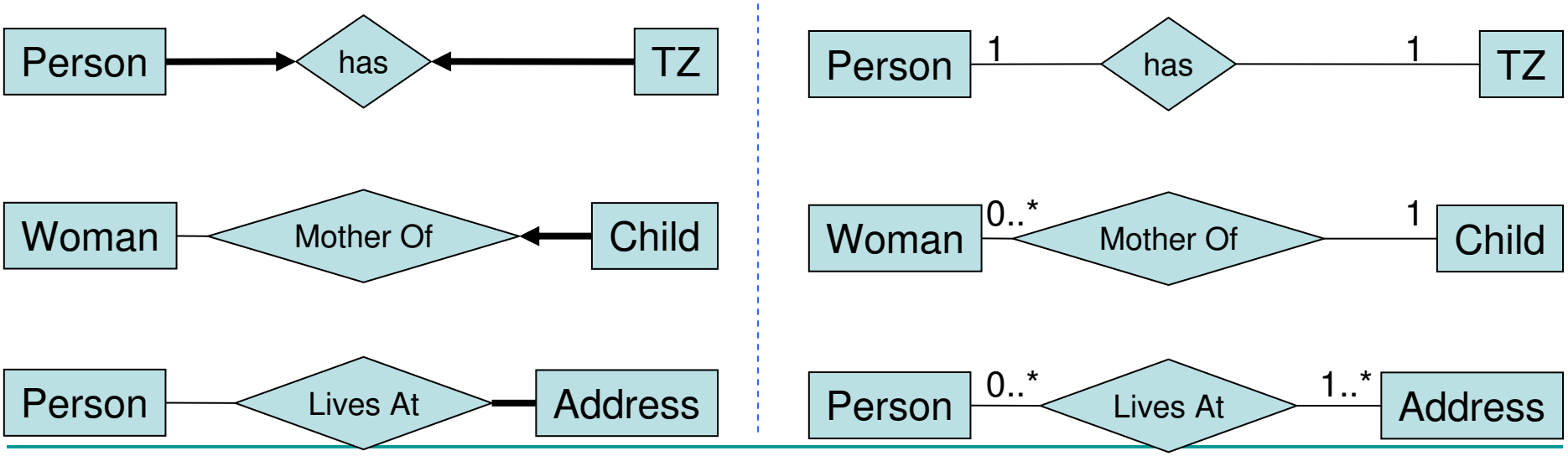


For a Team

- In a large organization, conceptual design is not done by one person
  - The team may not even be in the same location
  - They may not meet that often
- Usually separate teams will design separate aspects
  - But they all interact…
- This is why good planning **before** the design starts is essential
  - Determine central entities and relationships
  - Devise **interfaces** for the central entities or relationships ahead of time
  - Separation of responsibilities is key

Tools and Techniques

- I presented here a particular style of ERDs
  - They are graphically simple and easy to learn
  - There are many other styles and techniques for ERDs
- Key constraints and participation constraints are expressed with arrows and heavy lines
  - The Information Systems Engineering book uses another style which makes them numerically explicit



Tools and Techniques

- Another common tool for design is the Unified Modeling Language
  - UML is a collection of documentation techniques
  - It includes a format for Entity Relationship Diagrams
  - Differs slightly from what I've presented

- You'll have a course on UML next semester

So Far

- Entity Relationship Diagrams
  - Additional Features
- Conceptual Design using Entity Relationship Diagrams
  - for Large Enterprises
- Introduction to Relational Model

The Relational Model

- Due to Codd
- Tables are referred to as **Relations**
- Relations are built from a **Schema**
  - Schema describes what each row (**Records**) looks like
- Describes what each **field** is named and its type
  - We can define custom domains for a given built in type (later)
  - We can define custom types (next semester)

Example Schema

- Students (*sid:*string, *name*:string, *login*:string, *age:*integer, *gpa:*real)

Sid	*Name*	*Login*	*Age*	*Gpa*
53	Jones	ajones@cs	18	34.5
54	Jones	bjones@cs	30	91.3
12	Smith	smith@is	23	78.2
53	Cohen	cohen@math	19	80.2

Duplicates and Ordering

- In general, you can't have duplicate rows
  - Keys prevent this is most cases
  - How?

- Without keys, some commercial databases allow duplicates
  - Why?

- In general, the order of records does not matter
  - Why?
  - We **can** sort, if we want
  - We **can** ask do filtering based on position (next semester)

Domains

- Simple built in domains include:
  - Integers
  - Reals
  - Characters (length) / Varchar (length)
  - Date
  - Date Time
  - Boolean
  - Text
  - Binary Large Objects (BLOB)
- Can impose custom domain constraints

- Abstractly, a schema is then:

$$\{\langle f_1 : d_1, \ldots, f_n : d_n \rangle \mid d_1 \in Dom_1, \ldots, d_n \in Dom_n\}$$

Some Relational Terms

- **Degree** or **arity** of a relation – number of fields
- **Cardinality** of a relation – number of records
- **Relational Database** – collection of relations with distinct names
- **Relational Database Schema** – collection of schemas for the relations in a relational database

Conclusion

- Entity Relationship Diagrams
  - Additional Features
- Conceptual Design using Entity Relationship Diagrams
  - for Large Enterprises
- Introduction to Relational Model