Data modeling using the entity-relationship model

In this Chapter , we describe the main techniques for gathering and capturing information about what the users require of a database system.
One of the most difficult aspects of database design is the fact that designers , database designers and end-users tend to view data and its use in different ways.

Unfortunately unless we get a common understanding that reflects how the enterprise operates the design we produce will fail to meet the user requirements.

We need to get a model for communication that is non-technical and free of ambiguities.

Chapter 3 Objectives

- How entities, tuples, attributes and relationships among entities are represented in relational modeling.
- Data Models Notations (Chens and Crows Foot notation)
- Classification Of Attributes.
- How to identify candidate ,primary and foreign key and alternate keys .
- Combining the different datasets and represent them as a data model .

.

Entity-relationship model (ER model)

ER model:

• Is used to create a logical data model that reflects all the user data requirements.

•It includes detailed descriptions of

- •entity types,
- •relationships, and
- •Constraints.

•It is majorly used for communication with non-technical users

ERD Development Process

i. Talk with the users to identify the basic forms and reports
ii. Identify the data items to be stored and characteristics associated to the data items.
iii. Select the primary key for each entity
iv. Establish relationships between the entities
v. Identify any business constraint
vi. .Draw an Entity model

Data modeling using the entity-relationship model

1. **Entity .** Is a real-world object distinguishable from other objects. A person, event, or thing about which data is collected.

Eg. *student*, *employee*, *Project* etc.

- Must be multiple occurrences to be an entity

•**Strong entity**

- An entity type is **strong** if its existence does not depend on some other entity type

Weak entity: Is an entity that is dependant on the existence of another entity. It cannot be uniquely identified by its attributes alone. Does not have a key of its own - It's identified by being related to another (Strong) entity type using some of their attribute values.

It's main characteristics

- identifying owner
- Does not have a key of its own

2. Determine the Attributes

Every Entity has Attributes.

- Attributes are characteristics that allow users to describe an entity.

- **Attribute Domain**
  - **Set of allowable values for one or more attributes.**
    Example of attributes with attribute domain ????

- Eg An Entity student has the attributes.
  - Student number
  - Name
  - Date of Birth
  - Course Number

Concepts (you should understand)

Attribute classification

- Simple attribute
- Composite Attribute
- Single Valued Attribute
- multi valued attribute
- Derived Attribute

- Key attribute: Candidate Key, Primary Key , Alternate Key .

- Null values: not applicable, unknown, missing.

Attributes

- Simple Attribute
  - Attribute composed of a single component with an independent existence. Simple attributes cannot be further subdivided into smaller components. Eg age , marital status.

- Composite Attribute

  -Some attributes can be further divided to yield smaller components with independent existence.

  - Attribute composed of multiple components, each with an independent existence.

    Eg Address (163 main st, Glasgow, G11 9QX) can be divided in to Street, city and postcode , DOB

Attributes

- **Single-valued Attribute**
  - Attribute that holds a single value for each occurrence of an entity type. Eg ? ?
- **Multi-valued Attribute**
  - Attribute that holds multiple values for each occurrence of an entity type. Eg??
- **Derived Attribute**
  - Attribute whose value is calculated from other attributes. The derived attribute need not be physically stored within the database.
  - Eg Employee age can be found by computing an integer value difference of

    Current Date- DOB=

Keys

Certain attributes identify particular facts with an entity , these are known as KEY attributes.

- **Candidate Key**
  - Minimal set of attributes that uniquely identifies each occurrence of an entity type.

- **Primary Key**
  - Field that is selected to uniquely identify each occurrence of an entity type.

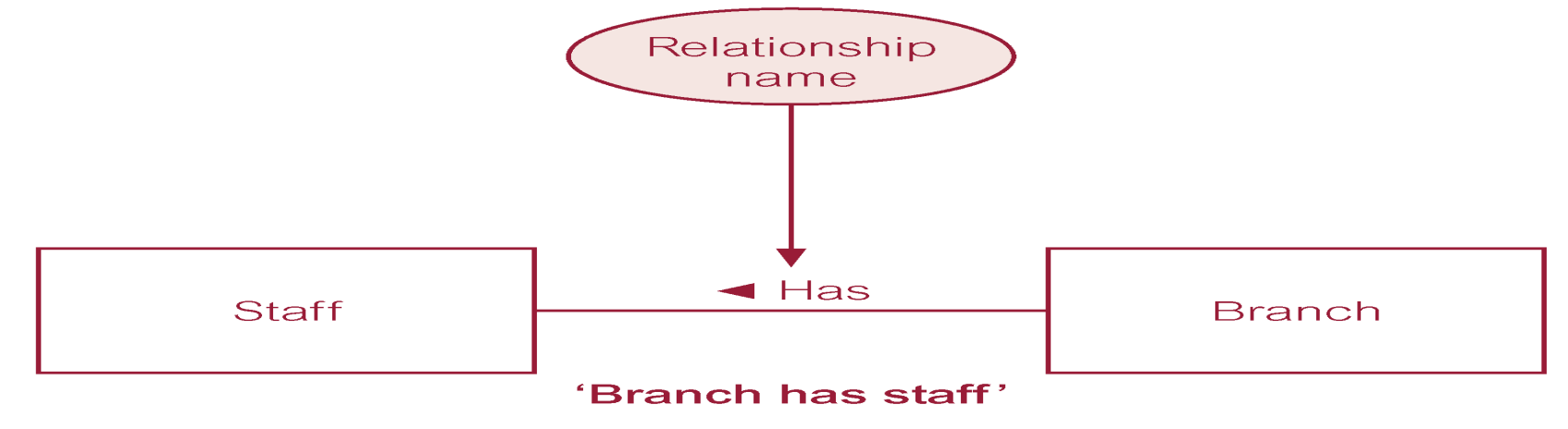
- **Alternate Key**

  Candidate keys that are not selected as the primary key

Naming and Defining Relationships

- A relationship is an association between entities .
- Relationship name is a verb phrase.

  Guidelines for defining relationships.
- Definition explains what action is being taken and why it is important.
- eg **Supervises or manages**



Relationship Types

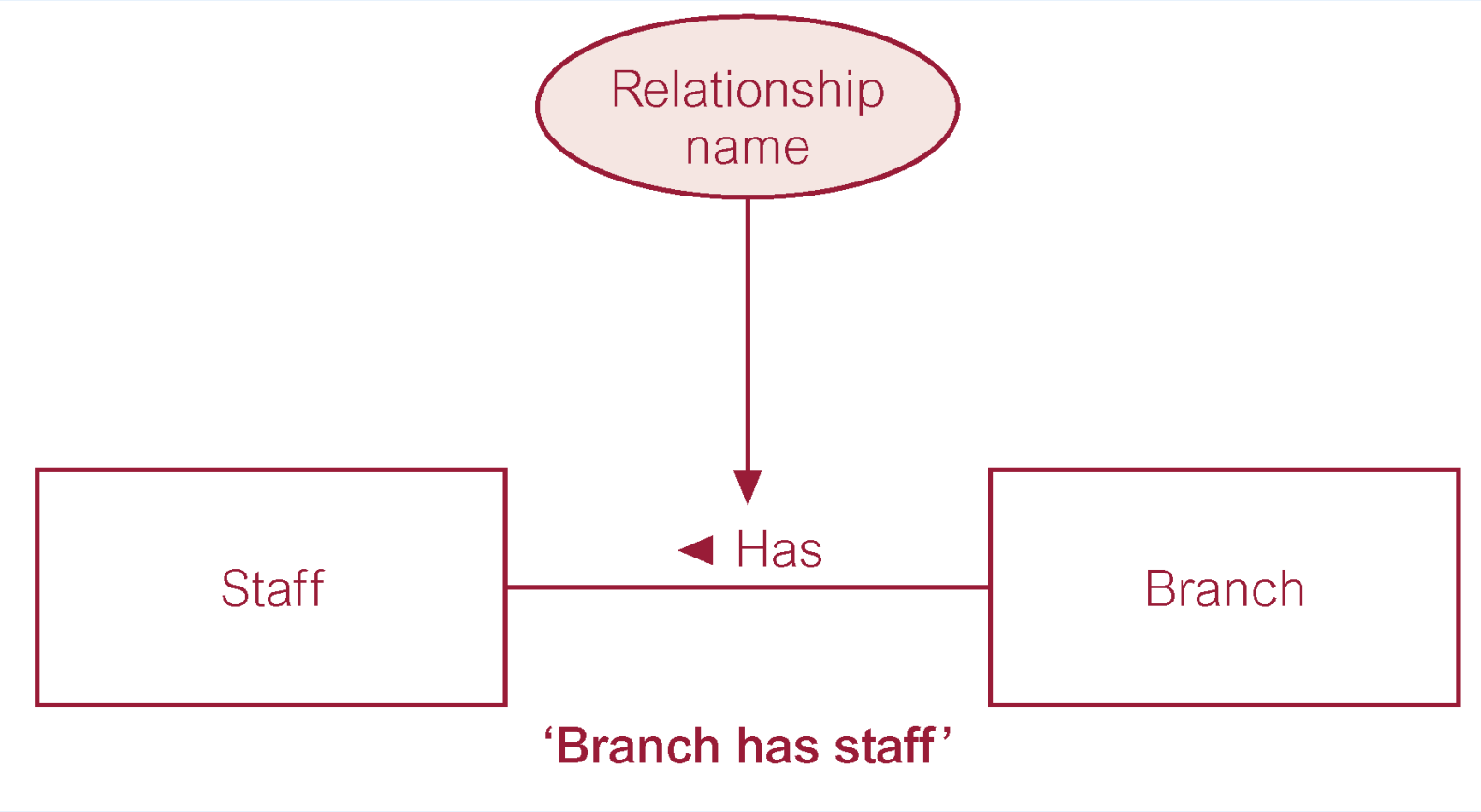
- **Degree of a Relationship**
  - **Number of participating entities in relationship.**
- **Relationship of degree :**
  - **two is binary**
  - **three is ternary**
  - **four is**

quaternary.

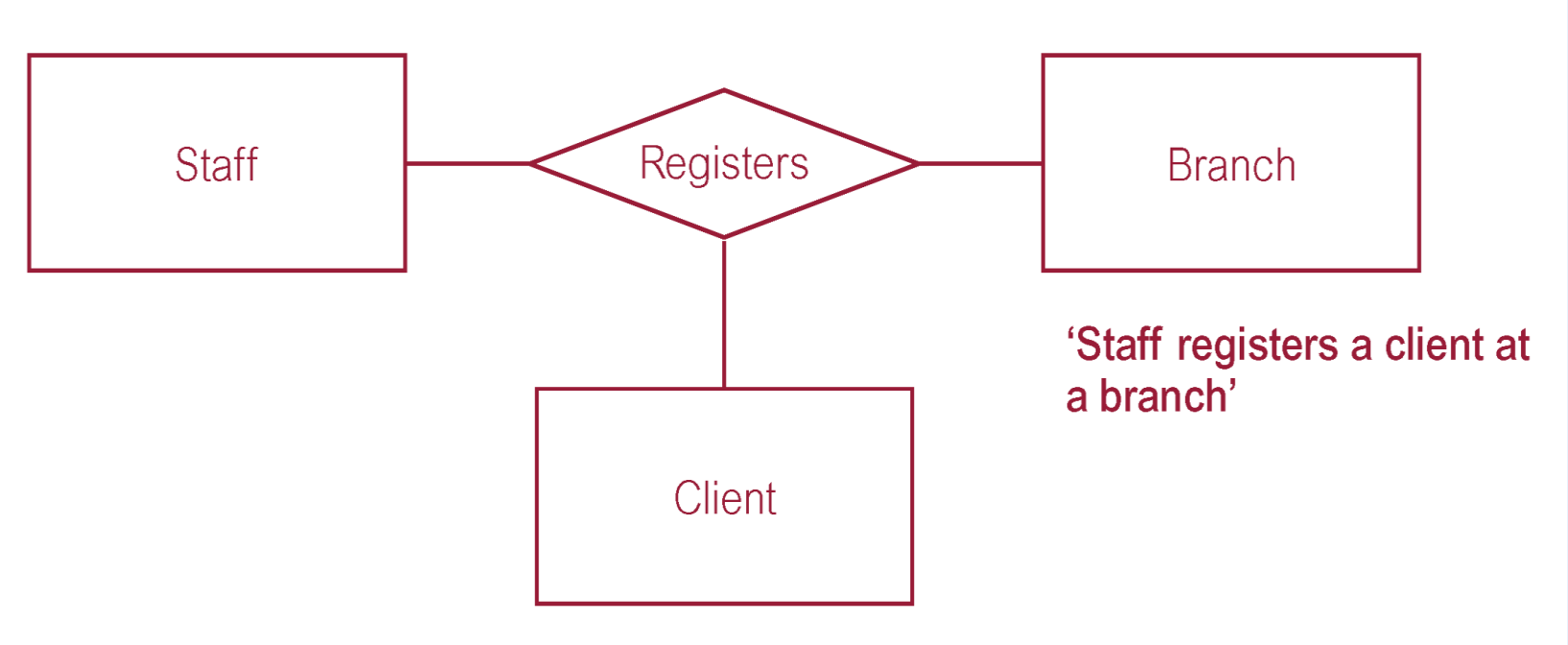
Relationship Types

- **Degree of a Relationship**
  - **Number of participating entities in relationship.**
- **Relationship of degree :**
  - **two is binary**
  - **three is ternary**
  - **four is quaternary.**

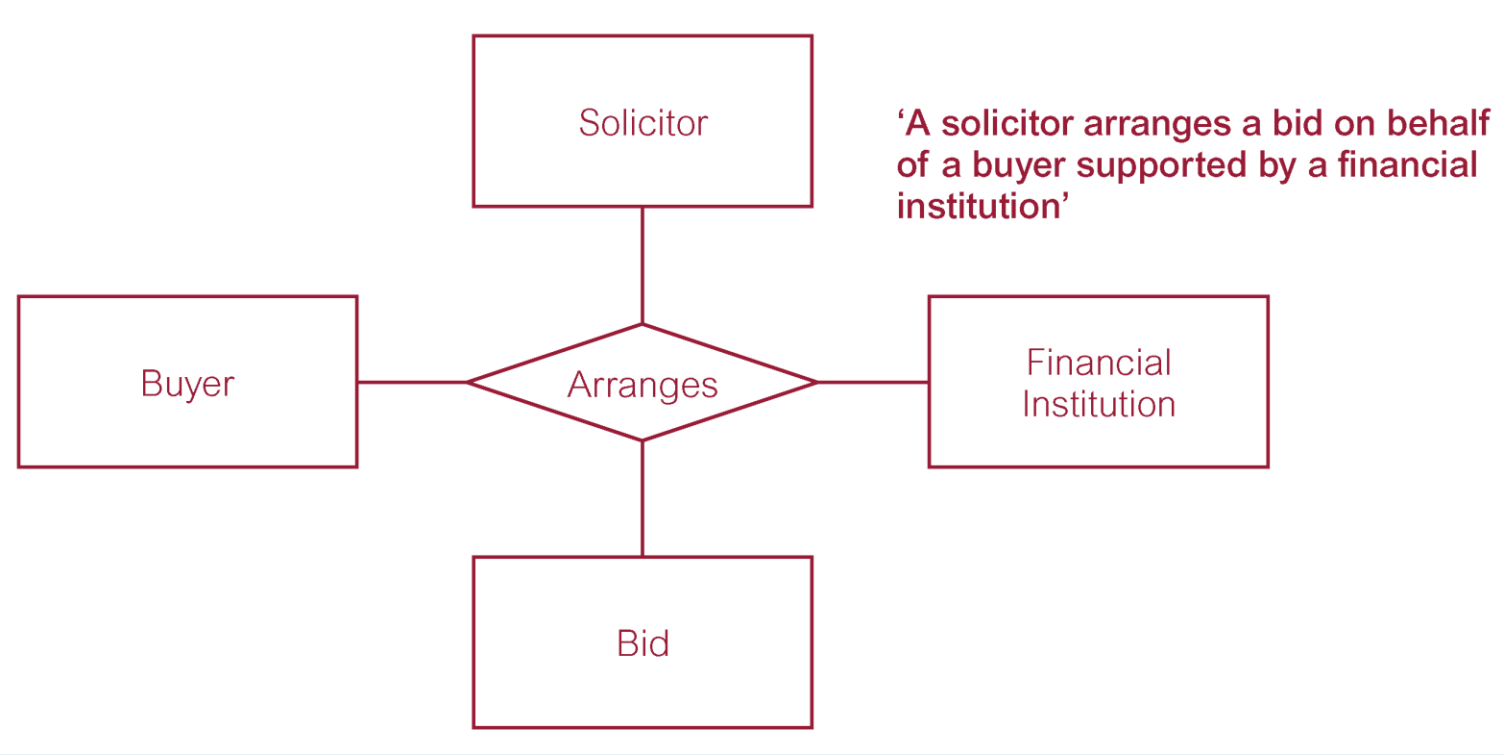
Binary relationship called Has



Ternary relationship called *Registers*



Quaternary relationship called *Arranges*



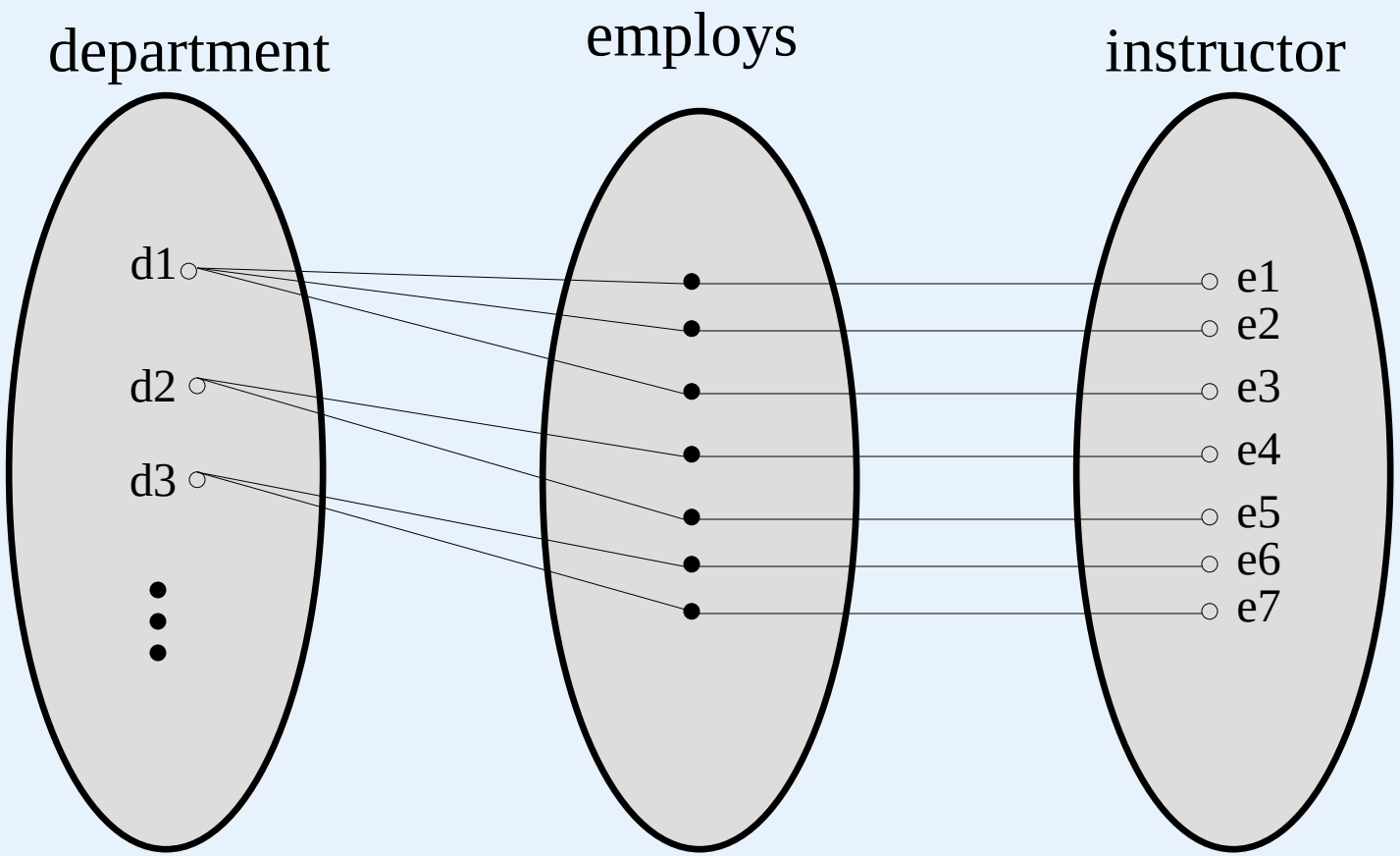
Multiplicity of a Relationship.

Describes maximum number of possible relationship occurrences for an entity participating in a given relationship type. In ERD multiplicity is indicated by placing the appropriate numbers beside the entities using the format (x,y).

x – minimum value, y – Maximum Value

Data modeling using the entity-relationship model

Consider some instances



- Many to Many relationship: Is a key constraint that indicates that many of one entity can be associated with many of another entity. An example of many to many relation is employees and their hobbies: A person can have many different hobbies and many people can have the same hobby.

Example

The company database keeps track of a company's employees, departments, and projects:

Requirements:

concerning the department:

1. company is organized into departments
2. a department has a unique name, a unique number, and a specific employee is its' manager
3. we track the start date for the manager function
4. a department may be in several locations
5. a department controls a number of projects

concerning the project:

6. a project has a unique name, a unique number, and a location

example continued

concerning the employee:

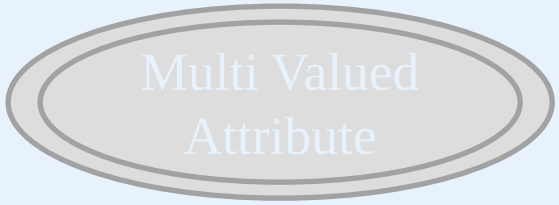
7. each employee has a name, social insurance number, address, salary, sex, and birth date
8. an employee is assigned to one department but may work on several projects which are not necessarily controlled by the same department
9. we track the number of hours per week that an employee works on each project
10. we keep track of the direct supervisor of each employee
11. we track the dependents of each employee (for insurance purposes)

concerning the dependent:

12. we record each dependent's first name, sex, birth date, and relationship to the employee

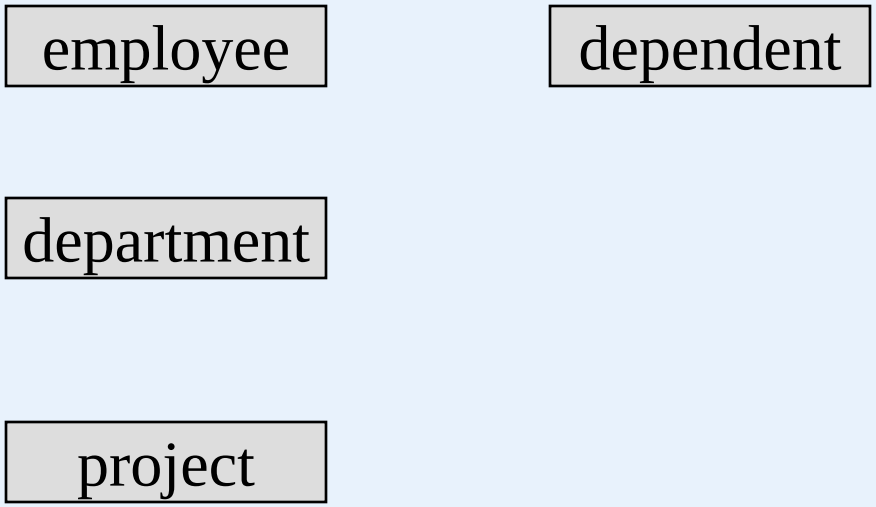
Basic ERD symbols

Derived Attribute

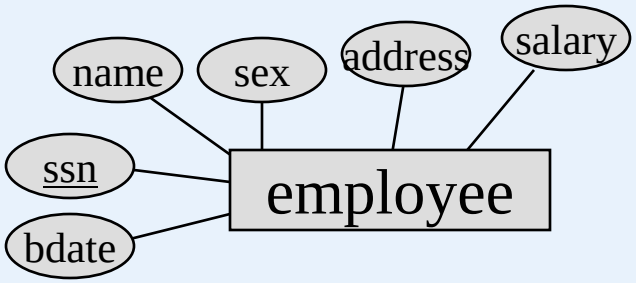


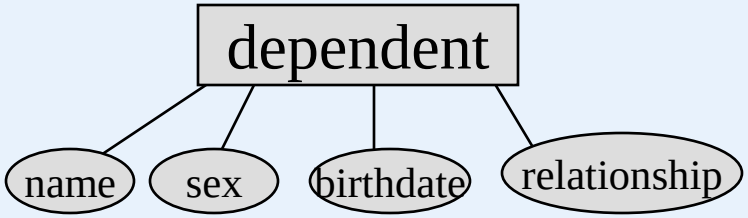
Derived attribute

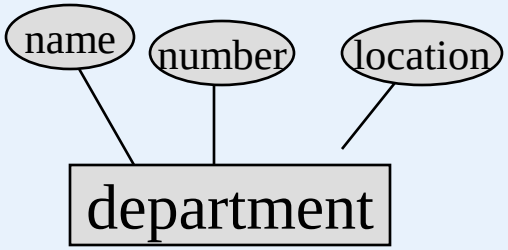
The entities/Relations:

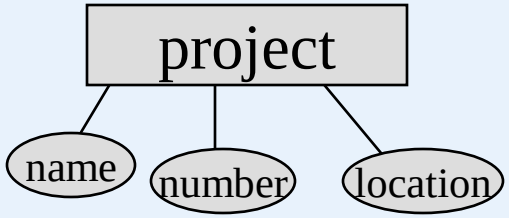


The entities/Relations:

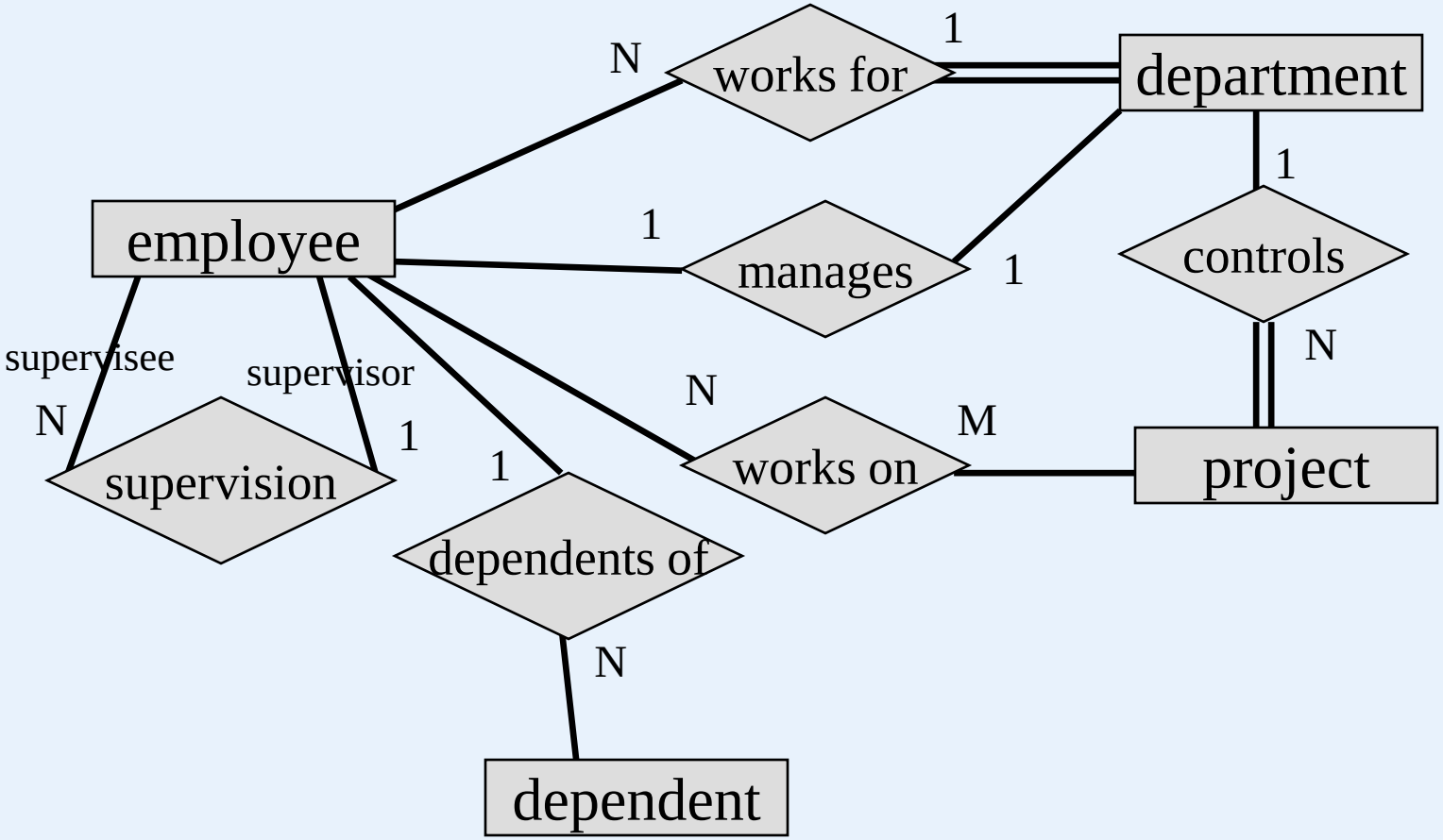








With relationships:



Concepts (you should understand)

Relationships

- Degree of a relationship
- Recursive relationship
- Constraints

  cardinality: m-n, 1-n, 1-1

Relationships are always labeled with **verb phrases.**

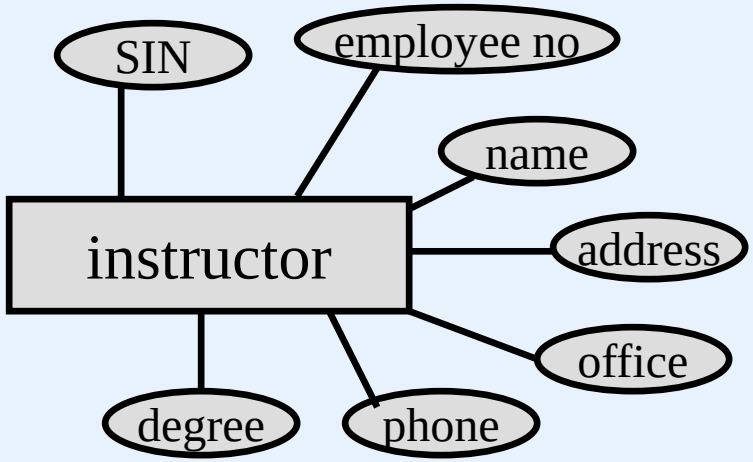
PROFESSOR teaches a CLASS,
BRANCH has STAFF.

Let’s consider a university environment:

At a high level, we need to:

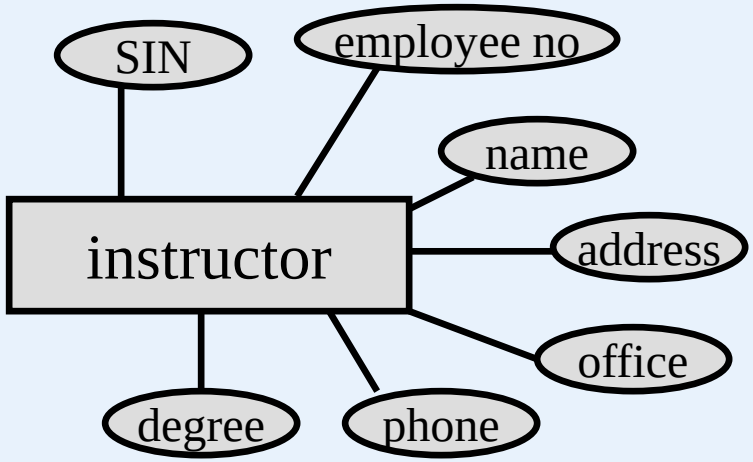
- offer courses
- register students
- assign instructors to courses
- assign grades
- anything else?

Instructors: let's assume this classification includes instructors, professors, part-time people (at least for now). These people have SINs (social insurance number), employee numbers, names, addresses, offices, phones, ...



Data modeling using the entity-relationship model

Is there a *key* attribute*?* What are the *domains?* Can any attribute be *null?* Is any attribute *composite*, *derived*, *multivalued?*

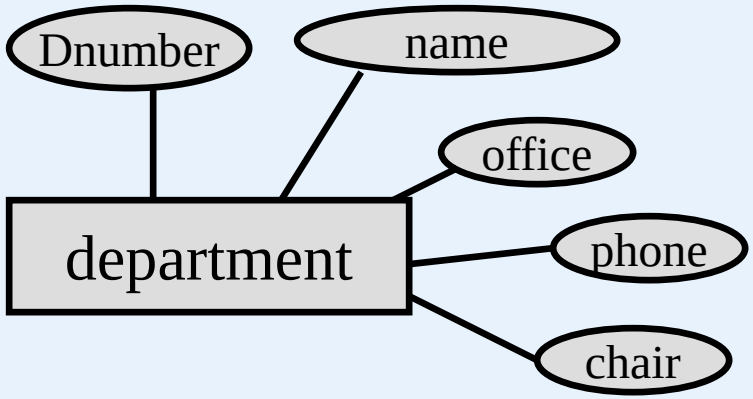


Is this a *weak* entity or a *strong* entity*?*

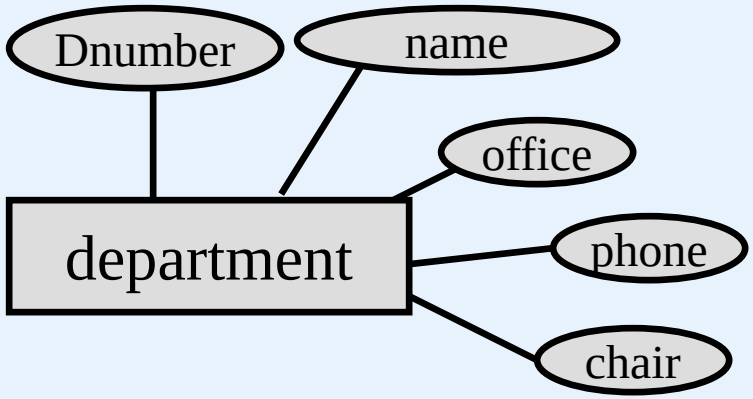
Should department be an attribute?

Departments: obviously instructors are employed by the University and associated with a department

A department has a name, number, office, chair(head of dept), ...

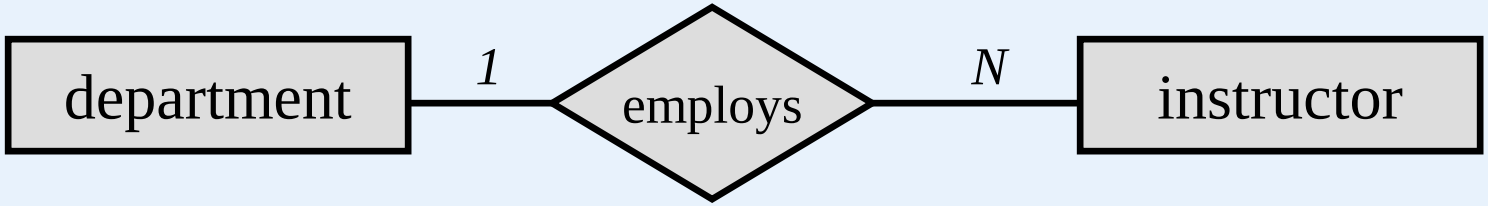


Is there a *key* attribute*?* What are the *domains?* Can any attribute be *null?* Is any attribute *composite*, *derived*, *multivalued?*



Should chair be an attribute, or is there a relationship between two entity types?

Employs relationship: If we assume the relationship between department and instructor is *1:N* . We associate any number of instructors with a single department



1:N is the *cardinality* of the relationship

the relationship is of *degree 2;* it is a *binary* relationship

CONCLUSION

The whole point about this chapter is “Given a dataset, how does one come up with (i.e. Design and formulate) entities and relationships and cardinalities between the different entities based on the business processes and needs of the organization for which they are designing the database.