ICCS240 Database Management

DB Design/ER

Motivating Scenario

You have been hired by a big purple bank to **design their online banking experience.**

In particular, the system must monitor:

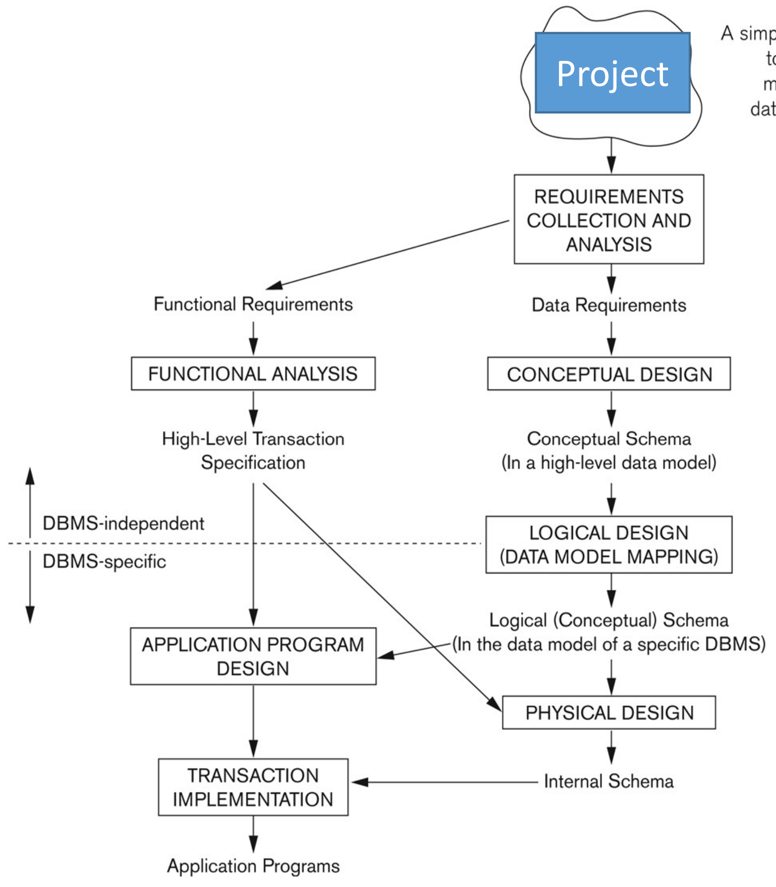
- Customers
- Accounts
- Loans
- Branches
- Transactions

How should you go about this?

Overview of the Process (Traditionally)

Two main activities:

- Database design
- Applications design



Project
REQUIREMENTS COLLECTION AND ANALYSIS
Functional Requirements
Data Requirements
FUNCTIONAL ANALYSIS
CONCEPTUAL DESIGN
High-Level Transaction Specification
Conceptual Schema (In a high-level data model)
DBMS-independent
DBMS-specific
LOGICAL DESIGN (DATA MODEL MAPPING)
Logical (Conceptual) Schema (In the data model of a specific DBMS)
APPLICATION PROGRAM DESIGN
PHYSICAL DESIGN
TRANSACTION IMPLEMENTATION
Internal Schema
Application Programs

Figure 3.1 A simplified diagram to illustrate the main phases of database design.

Figure - Copyright © 2007 Ramez Elmasr and Shamkant B. Navathei

For today:

- Conceptual design (via ER)

Next lecture: Convert that into relational schema

Summary: what questions to ask when designing a DB

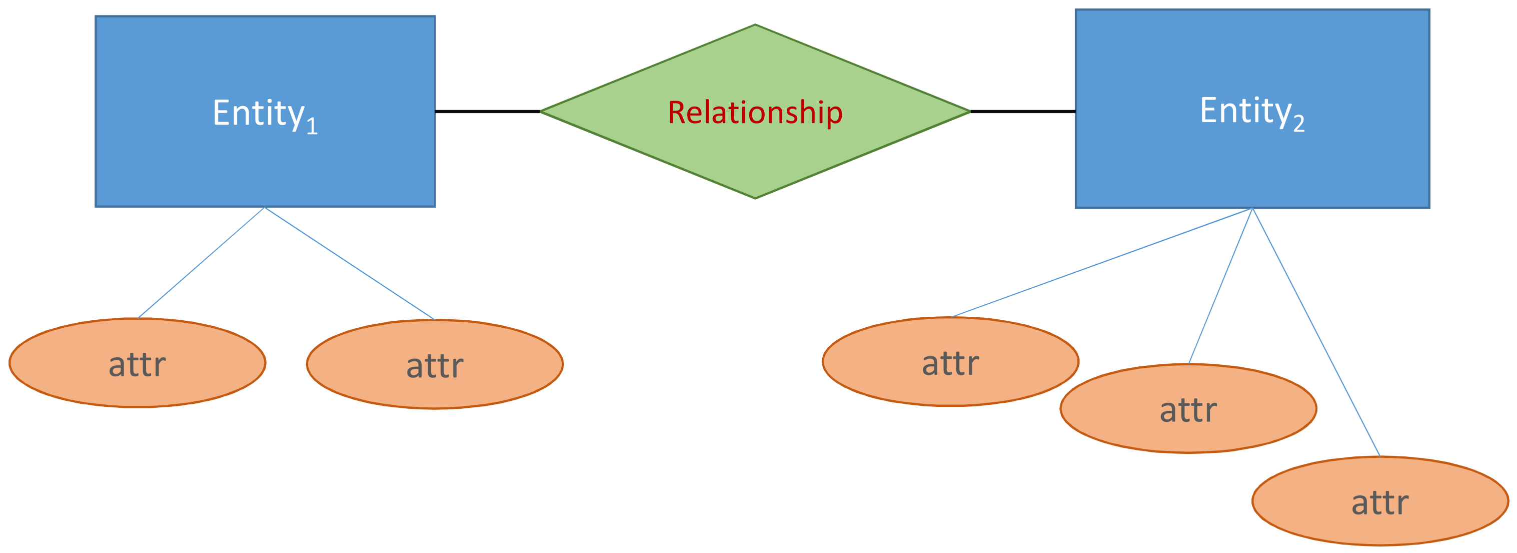
- What are the **entities** (objects, individuals, …)?
- Which **relationships** exist amongst entities?
- What information (**attributes**) do we want to store about these entities and relationships?
- What are the **integrity constraints**?

The answers can be represented in an

Entity Relationship Diagram (ER diagram)

Conceptual Design:
The Entity-Relationship (ER) Model

… provides a framework for thinking about data in terms of **entities** and their **relationships**.



Entities & Entity Sets/Types

- **Entity:** An object distinguishable from other objects (e.g., an employee)
  - An entity is described by a set of **attribute**s.

- **Entity Set/Entity Type:** A collection of similar entities (e.g., all employees)
  - All entities in an entity set have the same set of ***attributes***.
  - Each attribute has a ***domain***.
  - Each entity set has a ***key*** (i.e., one or more attribute whose values uniquely identify an entity)

Adapted from source: Werner Nutt

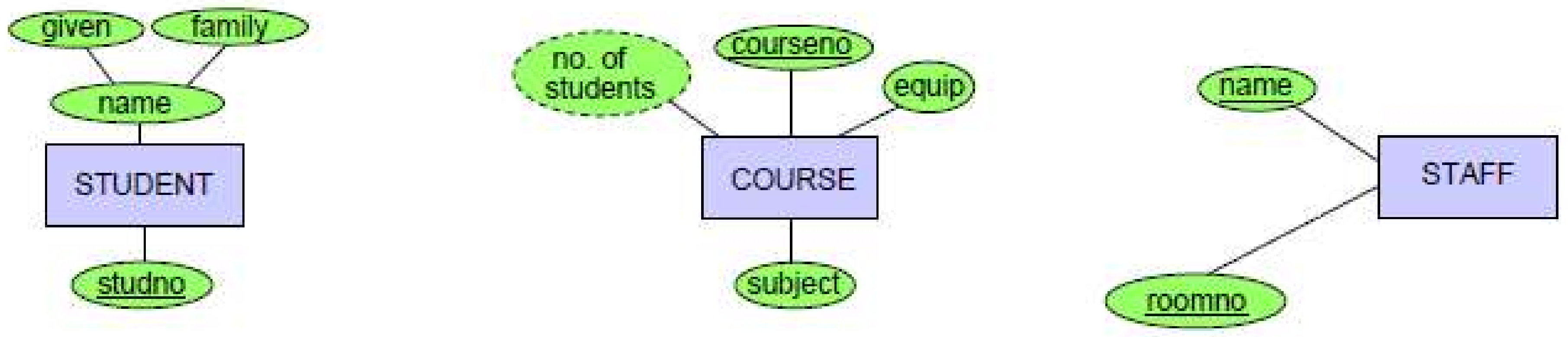
Attributes

- For every **attribute** we define
  - *domain* or *data type*
  - format: composite or atomic
  - whether it is **derived**
    (value is calculated from other attributes, e.g., average gpa)
  - whether it is **multi-valued**
    (multiple values for the attribute, e.g., a person may have more than 1 phone numbers)

- Every entity type must have as ***key*** an attribute or a set of attributes

Adapted from source: Werner Nutt

Graphical Representation of Entity Sets



- **Entity Sets** are drawn as rectangles
- **Attributes** are drawn using ovals
- *Composite* attributes combine two or more attributes
- *Derived* attributes are indicated by dashed ovals
- *Multivalued* attributes are indicated by double ovals
- The attributes making up the **key** are underlined

Adapted from source: Werner Nutt

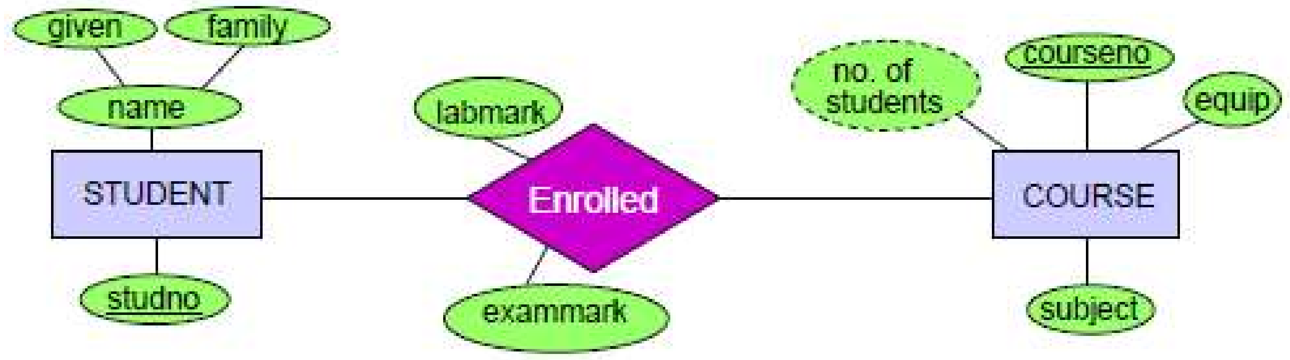
Relationships & Relationship Sets/Types

- **Relationship**: An association between two or more entities (e.g., Joe Smith is "enrolled" in CS123)
  - Relationships may have **attributes**.

- **Relationship Set/Entity Type**: A collection of similar relationships
  - An n-ary relationship type relates n entity types $E_1, \dots, E_n$
  - Each relationship involves n entities: $e_1 \in E_1, \dots, e_n \in E_n$

Adapted from source: Werner Nutt

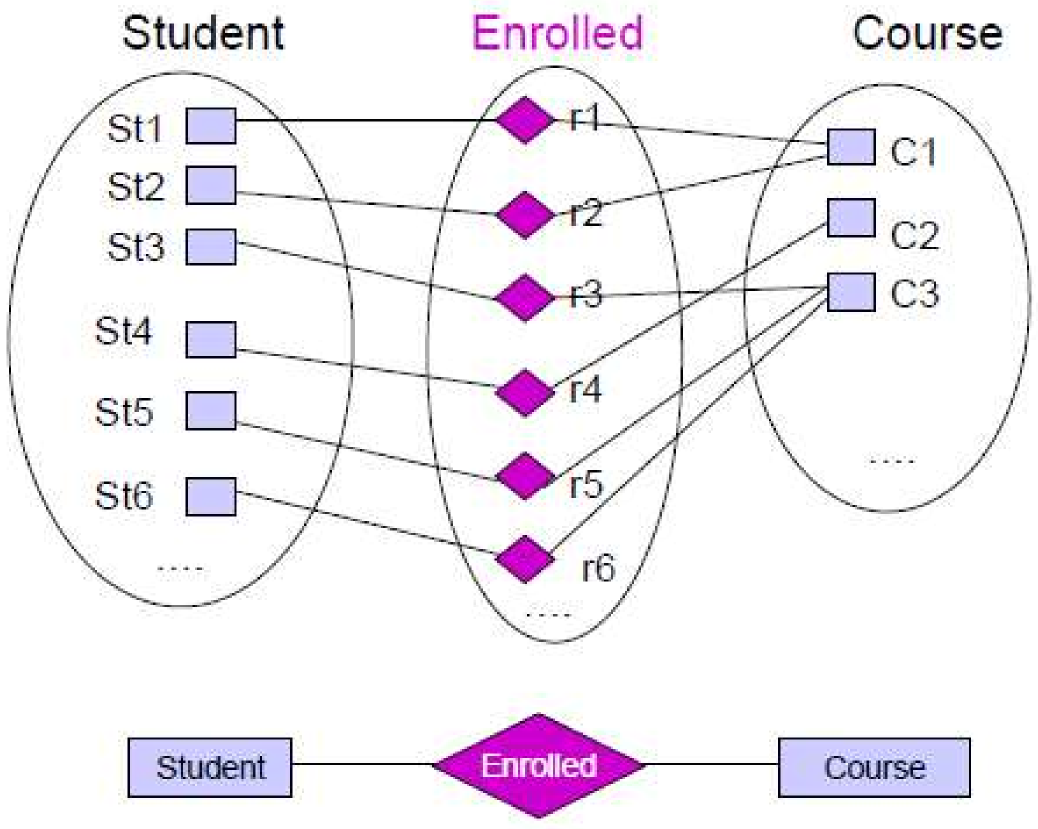
Graphical Representation of Relationship Types



- **Relationship** sets are drawn as diamonds

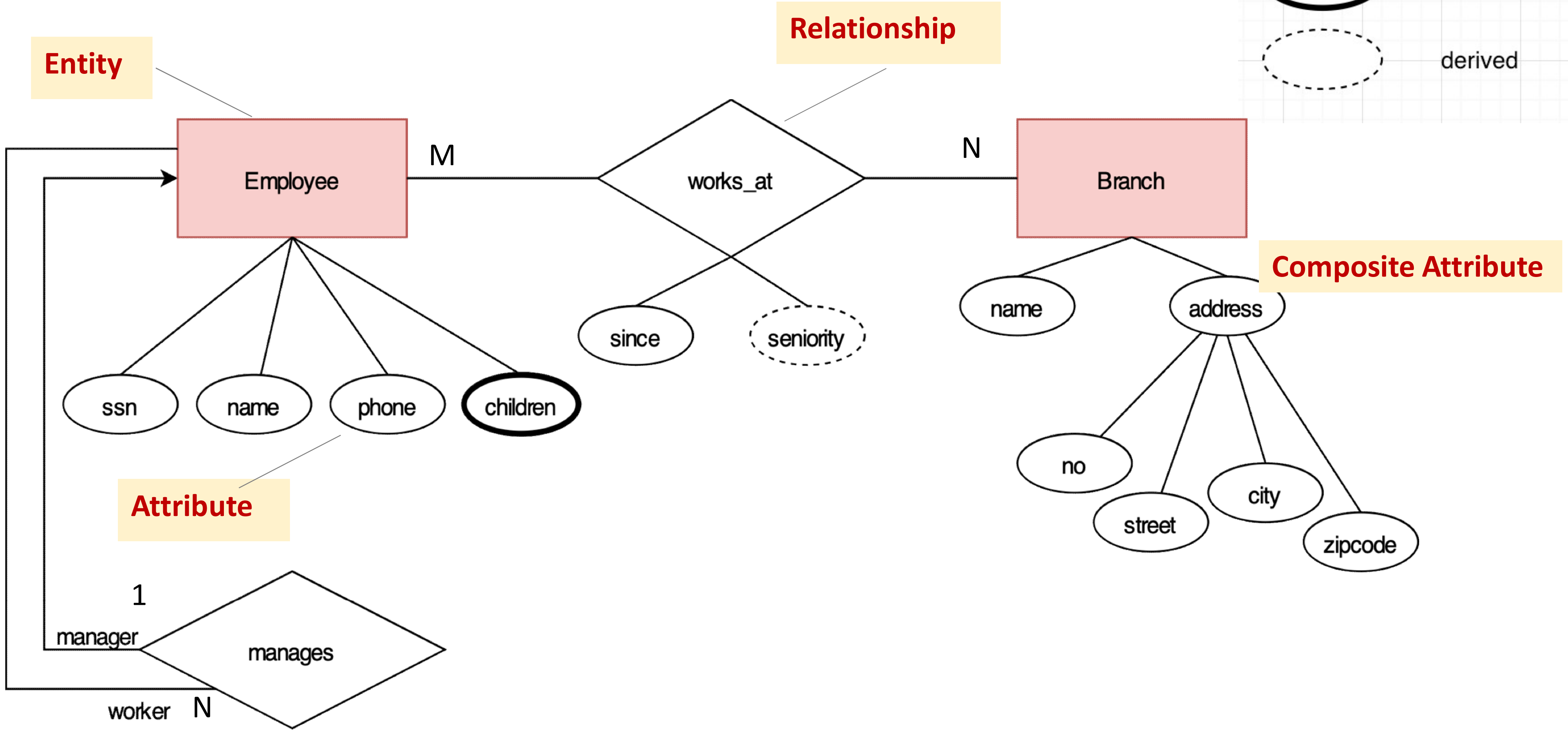
Adapted from source: Werner Nutt

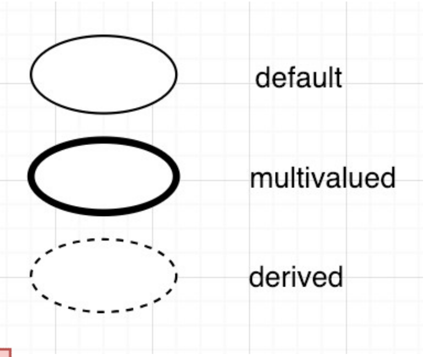
An Instance of a Relationship Type



source: Werner Nutt

Larger Example





Roles and Recursive Relationships

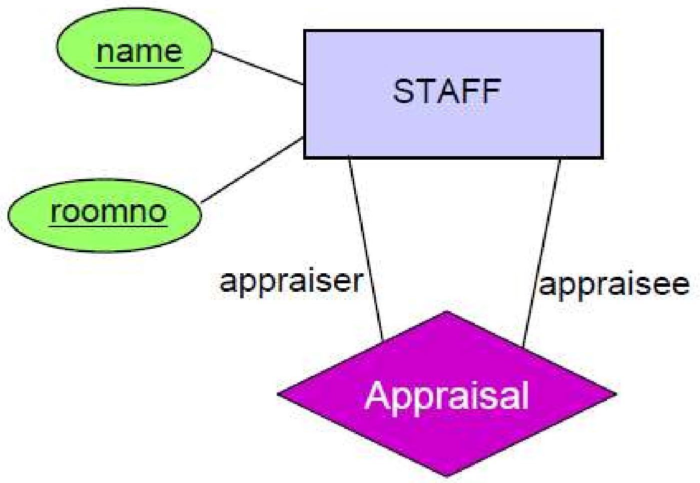
An entity type can

- participate in several relationship sets

and

- participate more than once in one relationship set

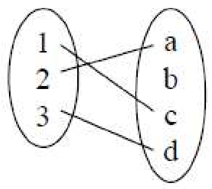
  (taking on different "roles")

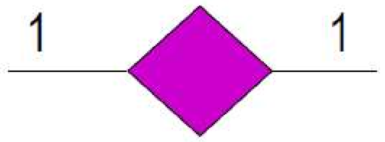


source: Werner Nutt

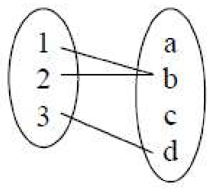
Multiplicity (cardinality) of Relationship Types

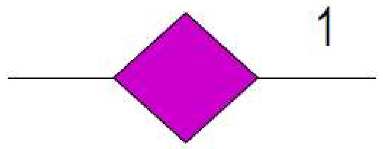
- one-one:

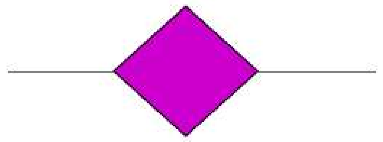




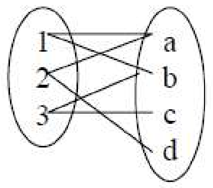
- many-one:







- many-many:



Sometimes the letters m, n are used to indicate the "many" side of relationships.

source: Werner Nutt

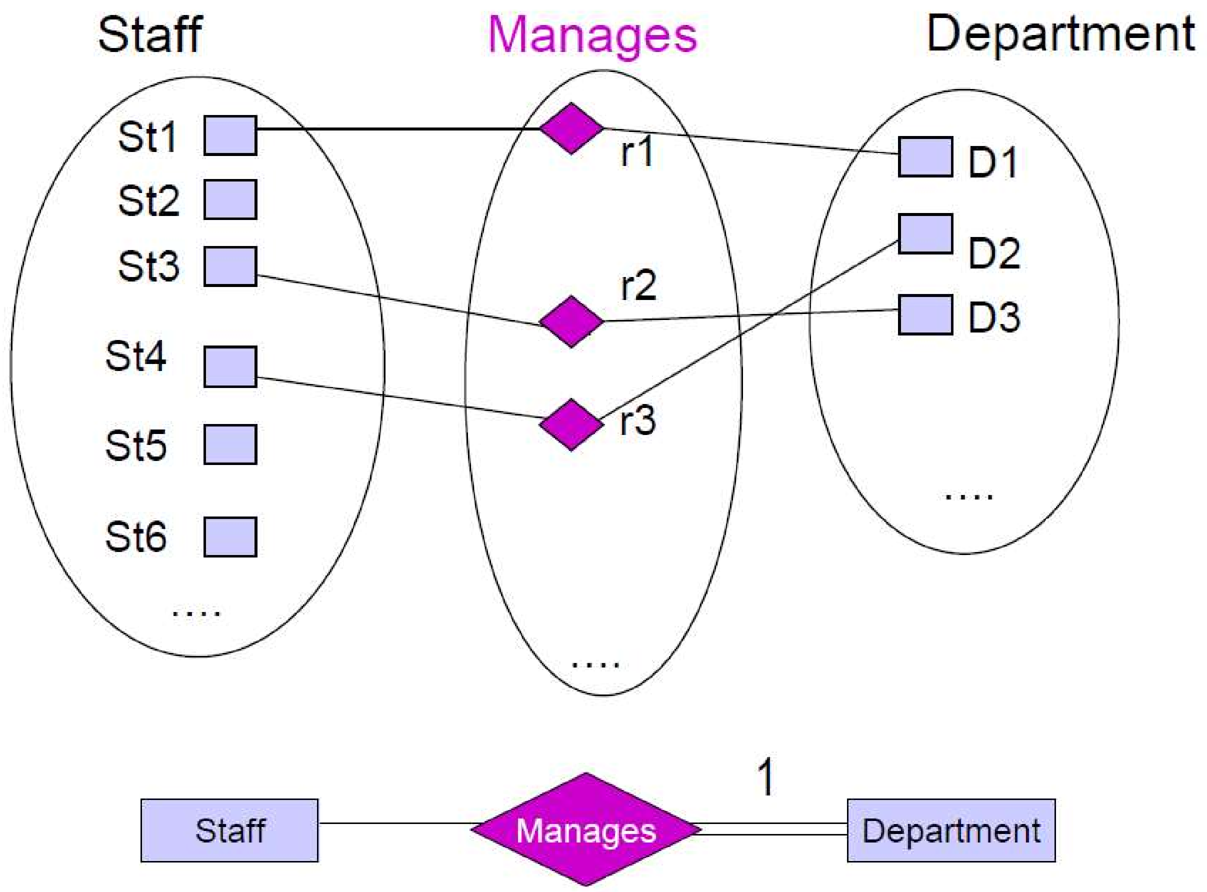
Participation Constraints

- Participation constraints specify whether or not an entity must **participate in a relationship** set
- When there is no participation constraint, it is possible that an entity will not participate in a relationship set
- When there is a participation constraint, the entity must participate *at least once*

- **Participation constraints** are drawn using a *double line* from the entity set to the relationship set

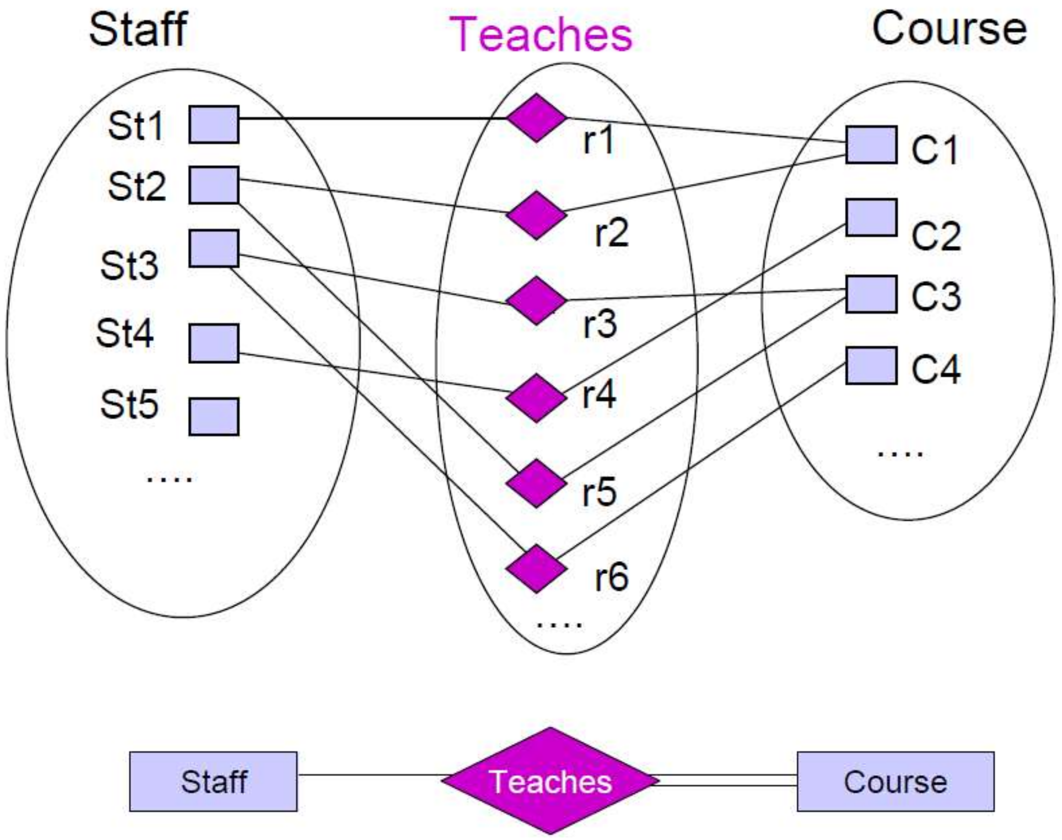
Adapted from source: Werner Nutt

Optional and Mandatory Participation



source: Werner Nutt

Many-Many Relationship Type
with optional and mandatory participation



source: Werner Nutt

Constraints

A **constraint** is an assertion about the database that must be true at all times.

Constraints are part of the database schema.

source: Werner Nutt

Modeling Constraints

Examples:

- **Keys**
  e.g., National ID / SSN / Passport Number uniquely identifies a person
- **Single-value constraints**
  e.g., a person can have only one father
- **Referential integrity constraints**
  e.g., if you work for a company, it must *exist* in the database
- **Domain constraints**
  e.g., peoples’ ages are between 0 and 150
- **Cardinality constraints**
  e.g., at most 100 students enroll in a course

source: Werner Nutt

Existence Constraints

Sometimes, the existence of an entity of type X
depends on the existence of an entity of type Y:

Examples:

- Book chapters presume the existence of a book
- Tracks on a CD presume the existence of the CD
- Orders depend on the existence of a customer

We call Y the *dominating* entity type and X the *subordinate* type

⇒ **strong** and **weak** entities

source: Werner Nutt

Strong and Weak Entities

Weak entities and **identifying relationships** are drawn using thick lines

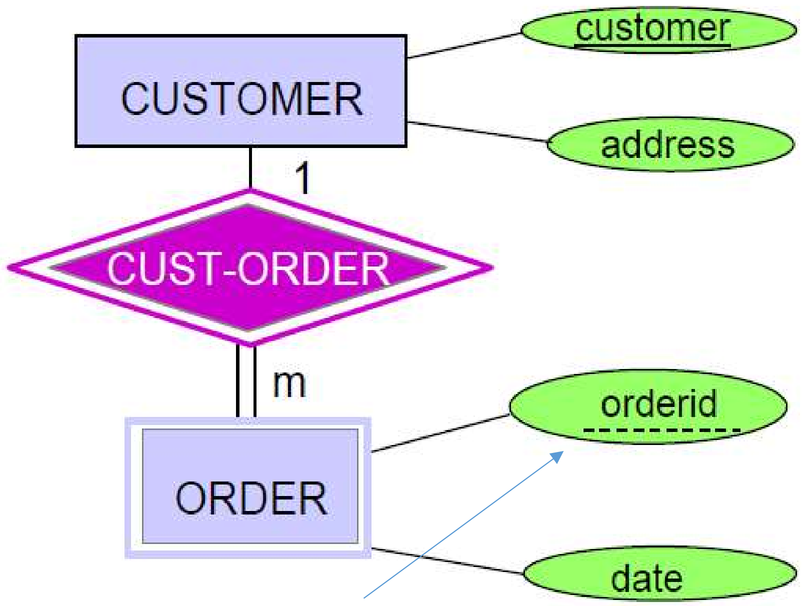
A *strong* entity type has an identifying primary key

A *weak* entity's key comes not (completely) from its own attributes,
but from the keys of one or more entities to which it is linked by a *supporting* many-one *relationship*

Identifying entity

Supporting, or identifying relationship

Weak entity

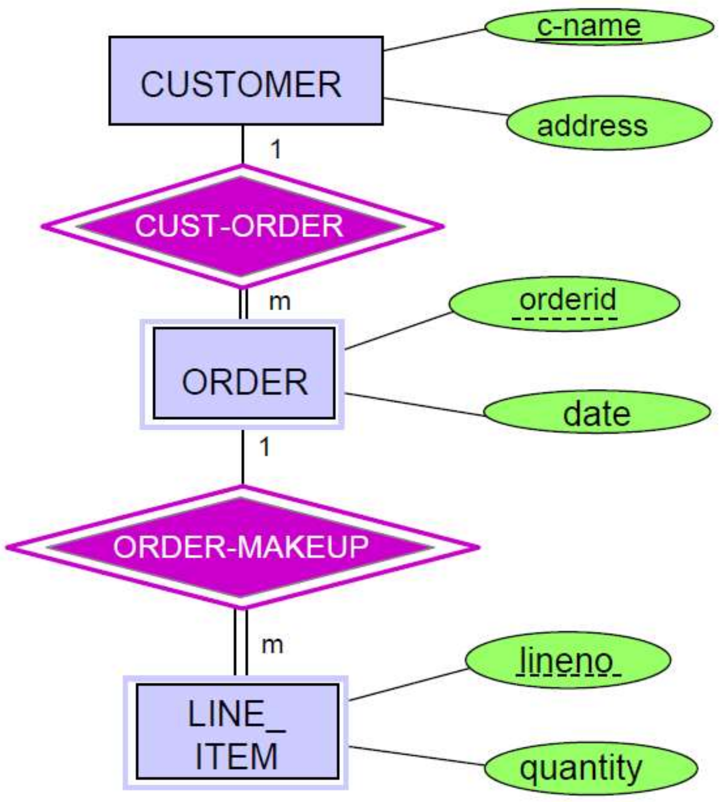


A *weak* entity type does not have its own primary key but does have a discriminator

So "customer" and "ordered" together are the primary key of ORDER

Adapted from source: Werner Nutt

Weak entities may depend on other weak entities



source: Werner Nutt

Even more sophisticated notations: Superclasses and Subclasses

A **subclass** entity type is a specialized type of a **superclass** entity type

A subclass entity type represents a subset or subgrouping of the superclass entity type’s instances

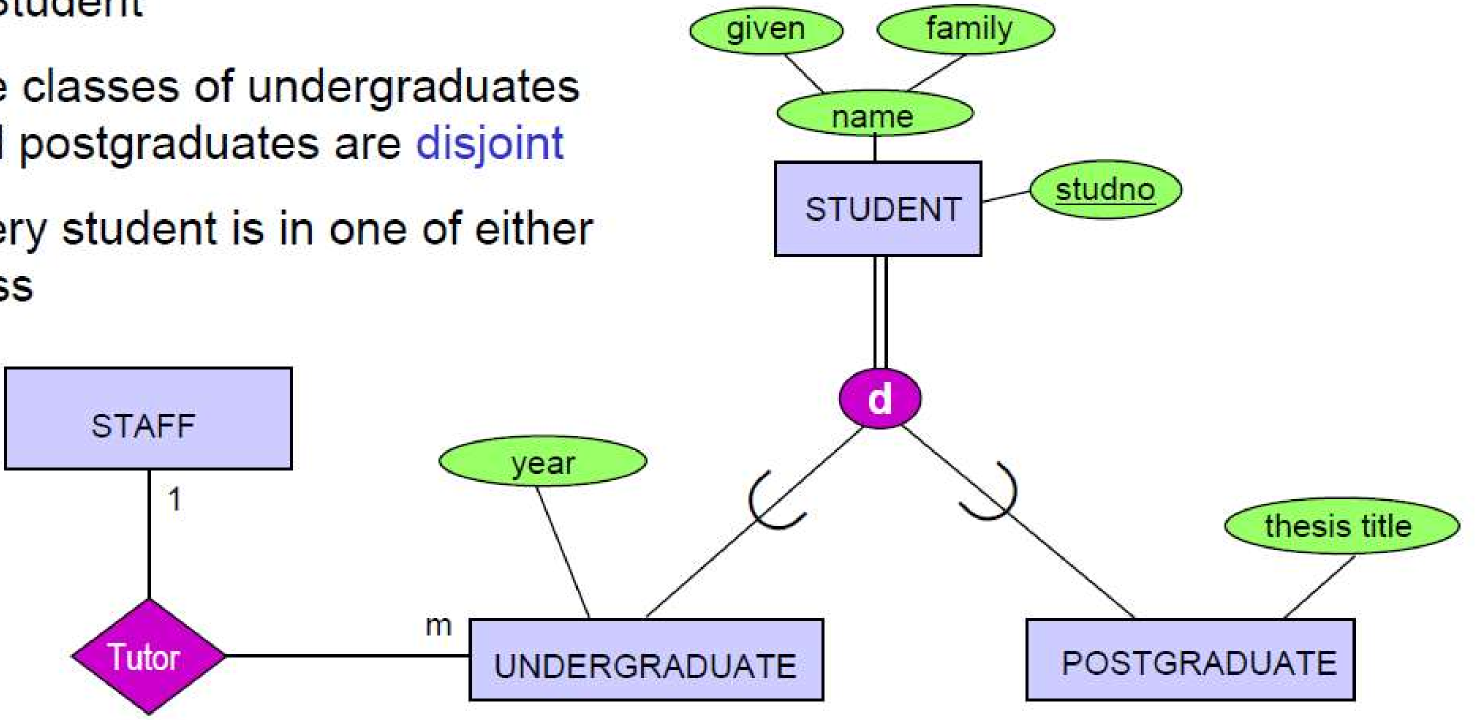
Example: Undergraduates and postgraduates are subclasses of student

Attribute Inheritance: Subclasses inherit attributes of their superclasses

Adapted from source: Werner Nutt

Example: students are undergraduates or postgraduates

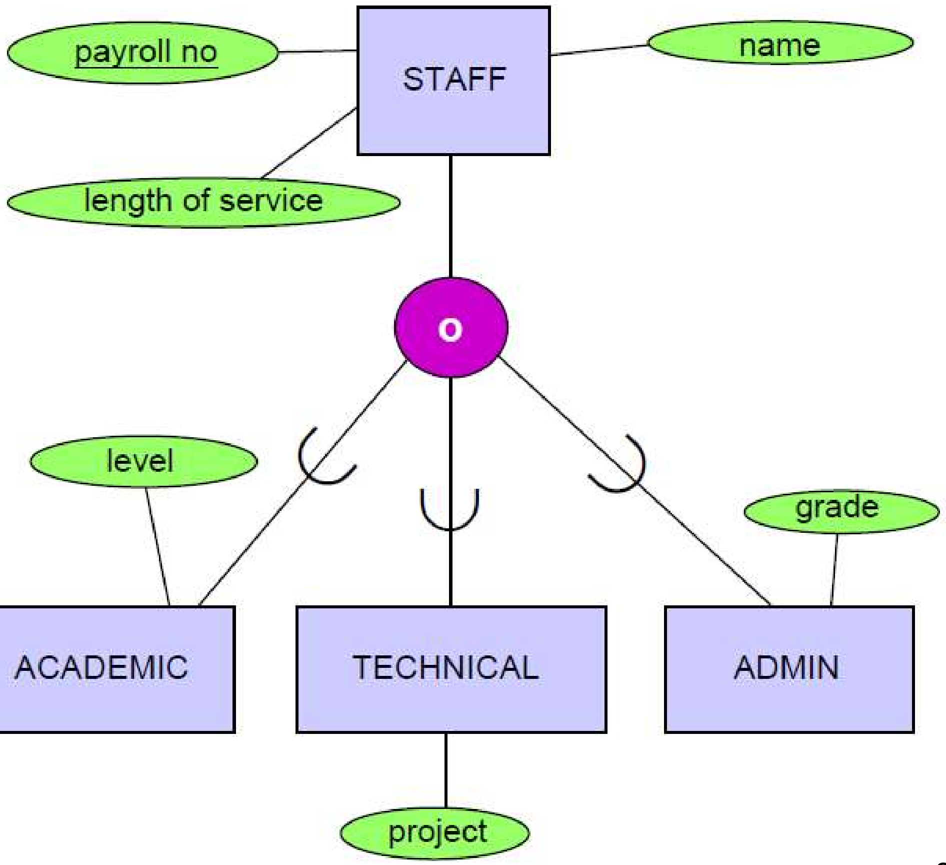
- Undergraduates and Postgraduates are subclasses of Student
- The classes of undergraduates and postgraduates are disjoint
- Every student is in one of either class



source: Werner Nutt

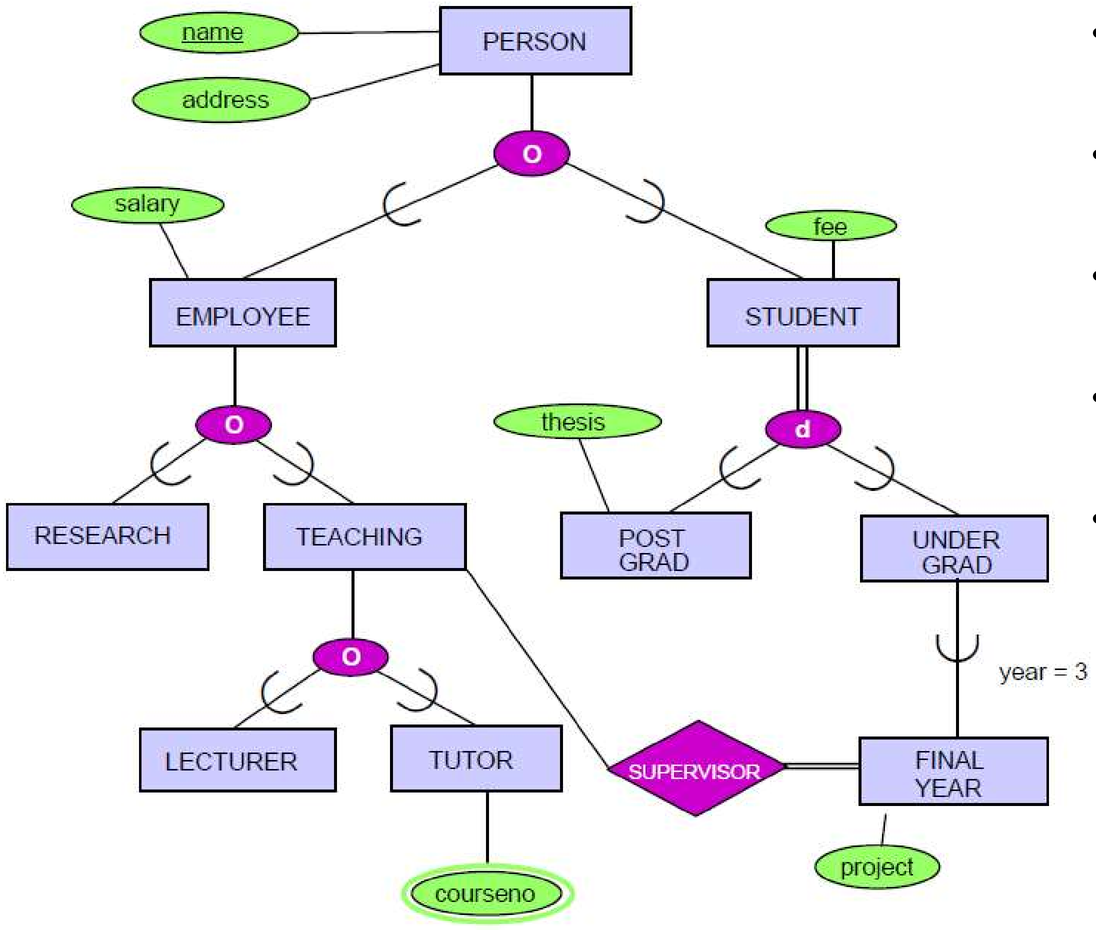
Example: subclasses of staff

- Academic, technical, and admin are three subclasses of staff
- The three classes may overlap



source: Werner Nutt

Larger example



- Every student is either a postgraduate student or an undergraduate student.
- A postgraduate student has a thesis title, on which he/she is working.
- An undergraduate student is in the final year, if he/she is in his/her 3rd year.
- Every final year student is working on a project.
- Every final year student is supervised by a member of the teaching staff.

source: Werner Nutt

Alternative Notations

(for your entertainment)

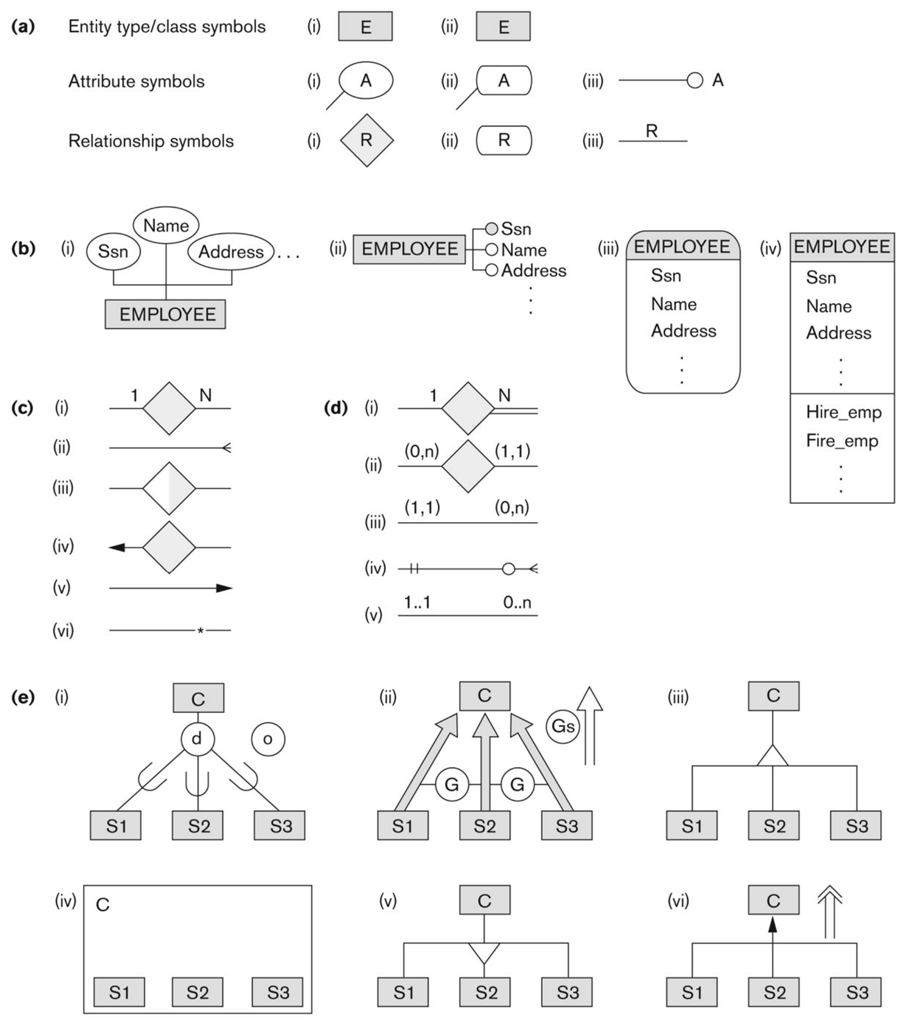


Figure A.1
Alternative notations. (a) Symbols for entity type/class, attribute, and relationship. (b) Displaying attributes. (c) Displaying cardinality ratios. (d) Various (min, max) notations. (e) Notations for displaying specialization/generalization.