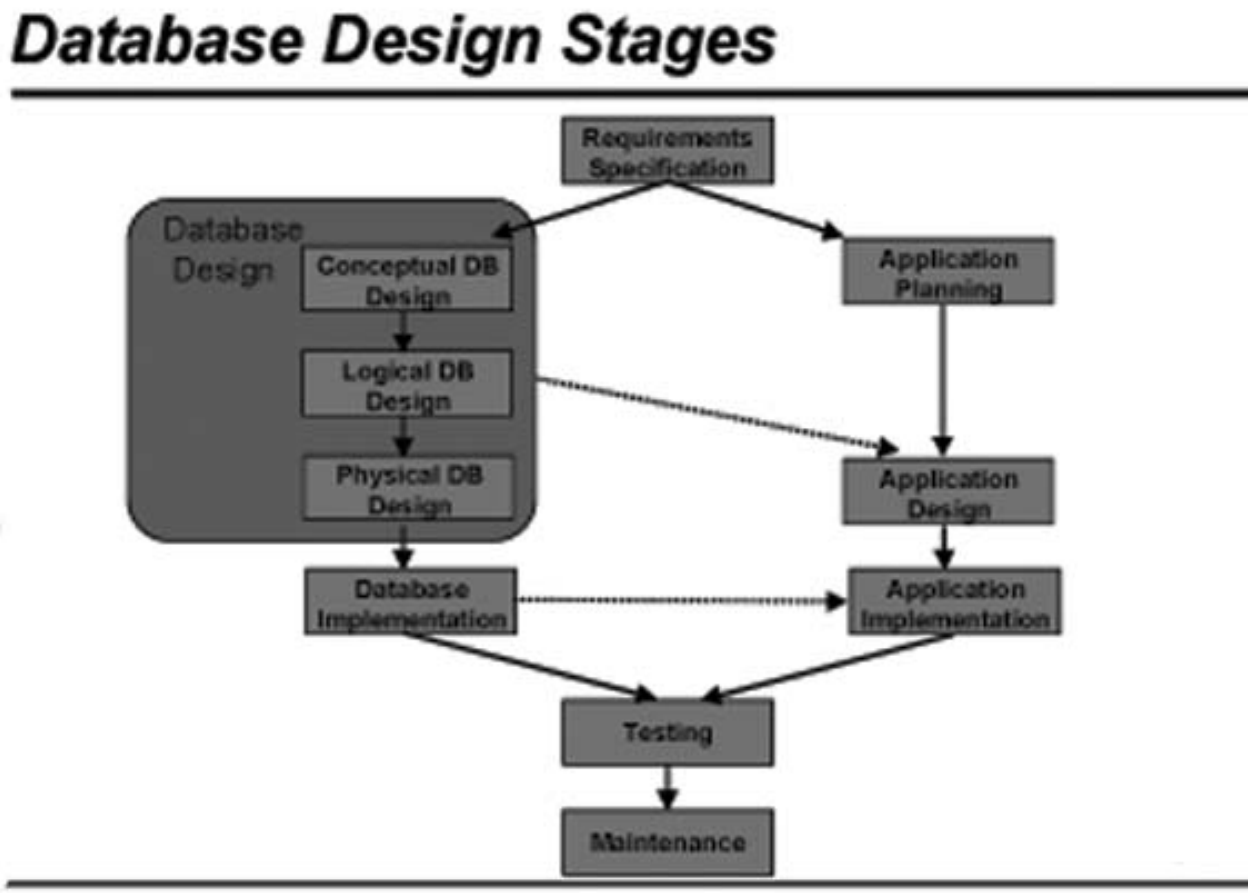
DATABASE DESIGN

- The ability to design databases and associated applications is critical to the success of the modern enterprise.
- Database design requires understanding both the operational and business requirements of an organization as well as the ability to model and realize those requirements using a database.
- Developing database and information systems is performed using a development lifecycle, which consists of a series of steps.



SPECIFICATION
REQUIREMENTS GATHERING

The most critical aspect of specification is the gathering and compilation of system and user requirements. This process is normally done in conjunction with managers and users.

The major goal in requirements gathering is to:

- collect the data used by the organization,
- identify relationships in the data,
- identify future data needs,
- and determine how the data is used and generated.

The starting place for data collection is gathering existing forms and reviewing policies and systems.

Then, ask users what the data means, and determine their daily processes. These things are especially critical:

- Identification of unique fields (keys)
- Data dependencies, relationships, and constraints (high-level)
- The data sizes and their growth rates

Fact-finding is using interviews and questionnaires to collect facts about systems, requirements, and preferences. Five fact-finding techniques:

- examining documentation
- interviewing
- observing the enterprise in operation
- research
- questionnaires

Database Design

The requirements gathering and specification provides you with a high-level understanding of the organization, its data, and the processes that you must model in the database.

Database design involves constructing a suitable model of this information. Since the design process is complicated, especially for large databases, database design is divided into three phases:

- Conceptual database design
- Logical database design
- Physical database design

CONCEPTUAL DATABASE DESIGN

Conceptual database design involves modelling the collected information at a high-level of abstraction without using a particular data model or DBMS.

REASONS FOR CONCEPTUAL MODELING

•Independent of DBMS.
•Allows for easy communication between end-users and developers.
•Has a clear method to convert from high-level model to relational model.
•Conceptual schema is a permanent description of the database requirements.

ENTITY-RELATIONSHIP MODEL

- Most popular conceptual model for database design
- Basis for many other models
- Describes the data in a system and how that data is related
- Describes data as *entities, attributes* and *relationships*

DATABASE REQUIREMENTS

• We must convert the written database requirements into an E-R diagram
• Need to determine the entities, attributes and relationships.

– nouns = entities
– adjectives = attributes
– verbs = relationships

Teaching Database

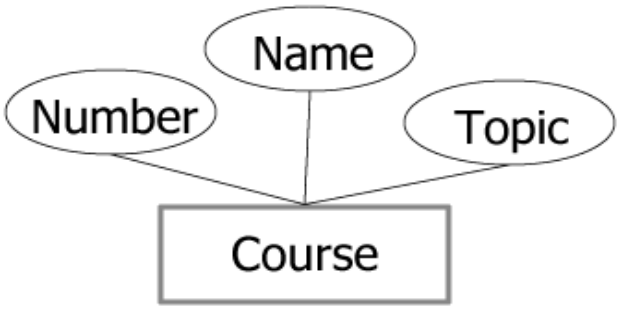
Design an E-R schema for a database to store info about professors, courses and course sections indicating the following:
• The name and employee ID number, salary and email address(es) of each professor
• How long each professor has been at the university
• The course sections each professor teaches
• The name, number and topic for each course offered
• The section and room number for each course section
• Each course section must have only one professor
• Each course can have multiple sections

Entities

- Entity – basic object of the E-R model
  - Represents a “thing” with an independent existence
  - Can exist physically or conceptually
    - a professor, a student, a course
- Entity type – used to define a set of entities with the same properties.

Entity and Entity Types

Entity Type



Entity

Number: 3753
Name: Database Management Systems
Topic: Introduction to DBMSs

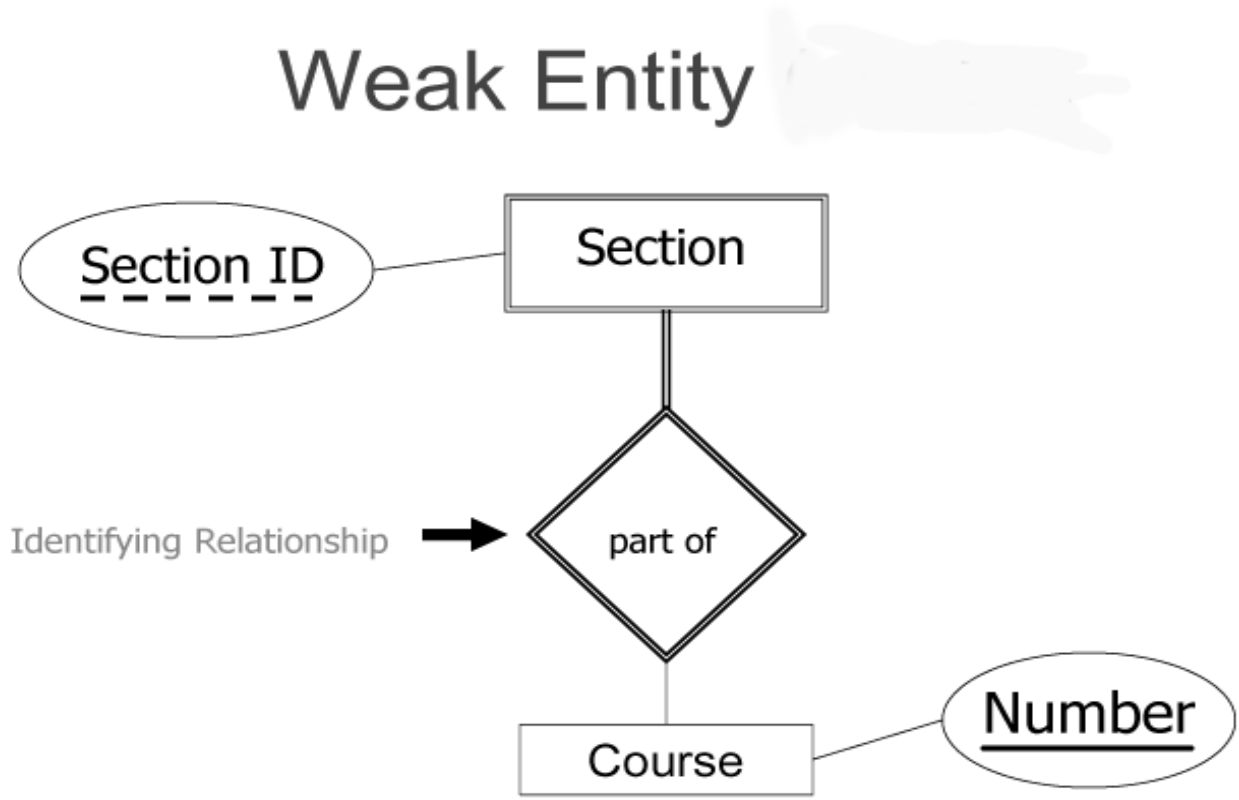
WEAK ENTITY

• Weak entities do not have **key** attributes of their own.

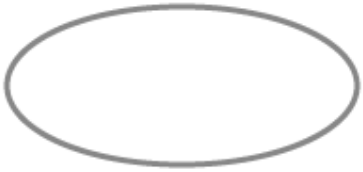
• Weak entities cannot exist without a relationship to another entity.

• A **partial key** is the portion of the key that comes from the weak entity. The rest of the key comes from the other entity in the relationship.

• Weak entities *always* have total participation as they cannot exist without the identifying relationship.



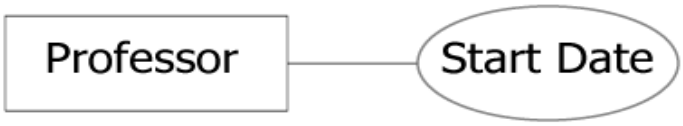
Attributes



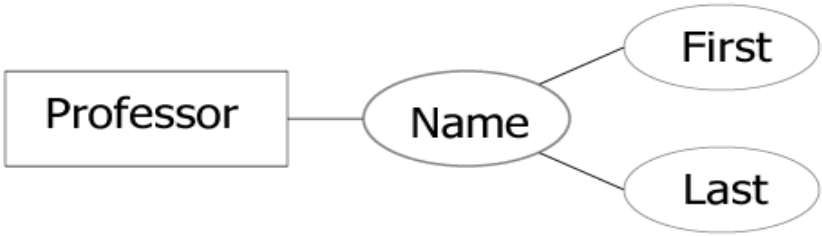
- Each entity has a set of associated properties that describes the entity. These properties are known as ***attributes***.
- Attributes can be:
  - Simple or Composite
  - Single or Multi-valued
  - Stored or Derived
  - NULL

Attributes

Simple



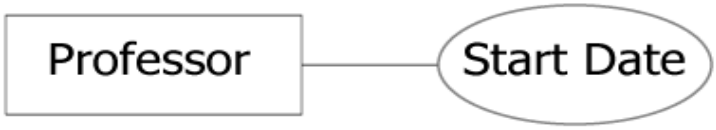
Composite



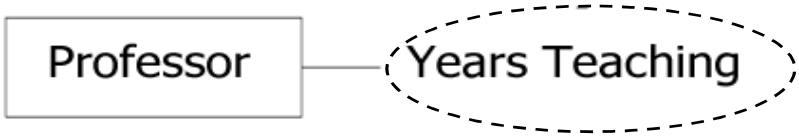
Single

Multi-Valued

Stored



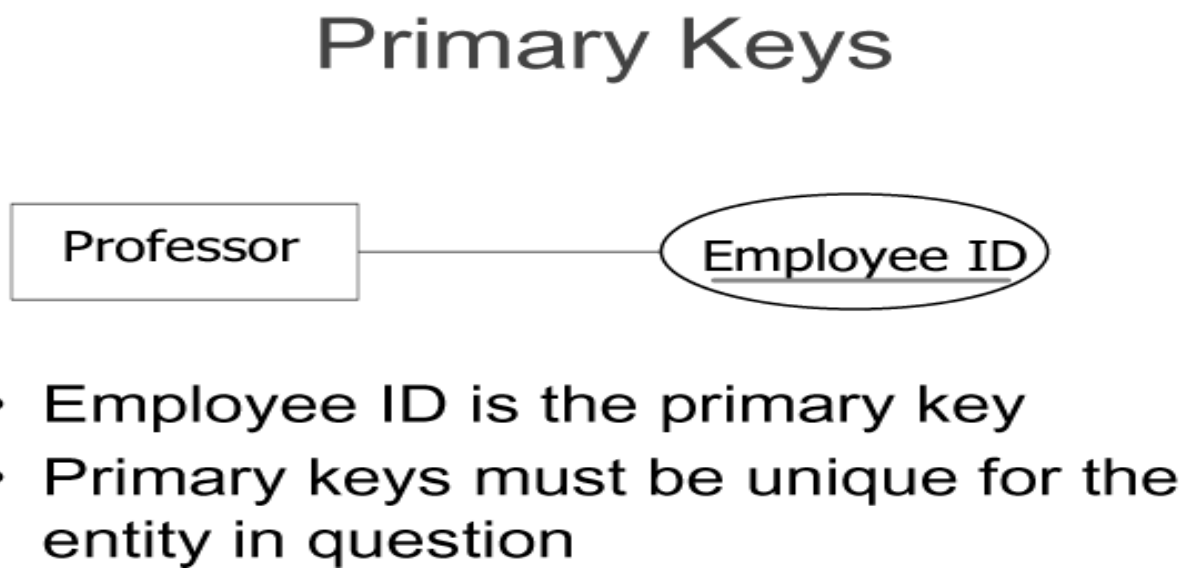
Derived



KEYS

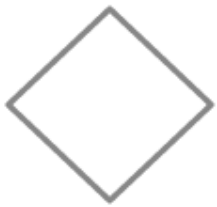
Candidate key: an attribute or set of attributes that uniquely identifies individual occurrences of an entity type.

Primary key: an entity type may have one or more possible candidate keys, one of which is selected to be a primary key.

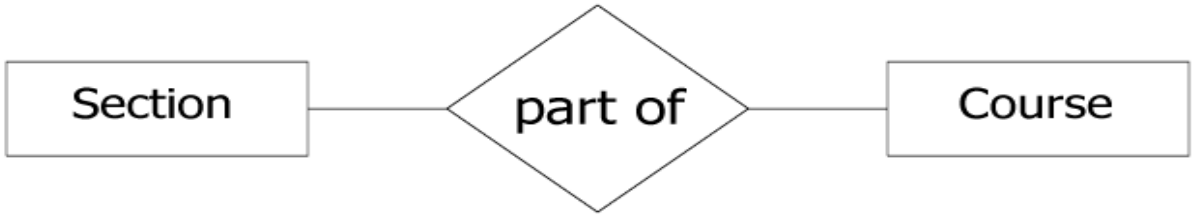


Composite key: A candidate key that consists of two or more attributes.

Relationships



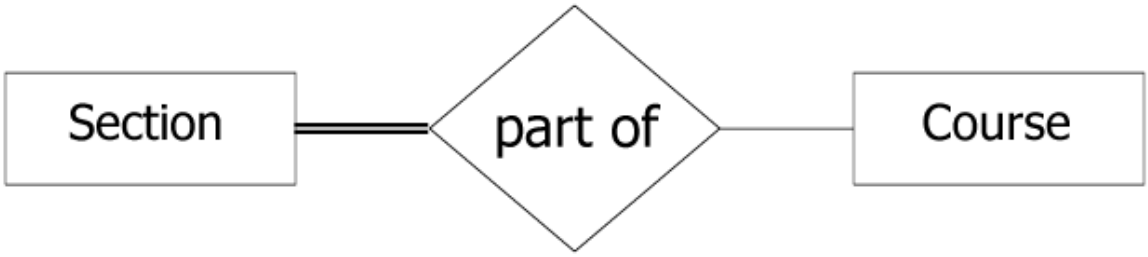
- defines a set of associations between various entities
- can have attributes to define them
- are limited by:
  - Participation
  - Cardinality Ratio



Degree of relationship: the number of participating entities in a relationship, e. g. binary, ternary etc.

Participation

- Defines if the existence of an entity depends on it being related to another entity with a relationship type.
  - Partial
  - Total



Cardinality

- The number of relationships that an entity may participate in.
  - 1:1, 1:N, N:M, M:1

Review of the E-R Diagram