INTRODUCTION TO CONCEPTUAL DATA MODELING

Objective

- Introduction +
- Database Design Process +
- Requirements +
- Conceptual Design +
- Entity-Relationship Model +

- Introduction

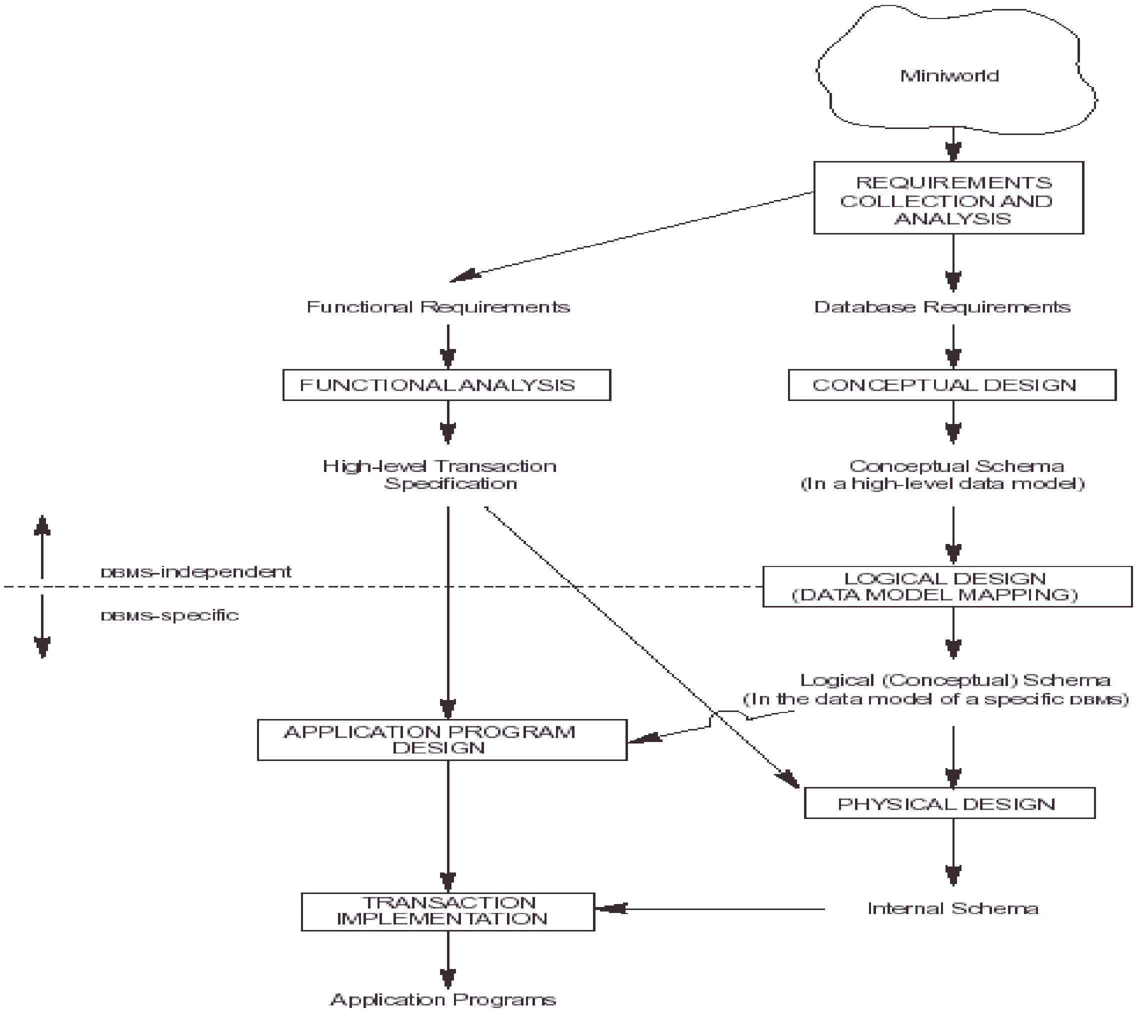
- A database application refers to a particular database and the associated programs that implement the database queries and update.
  - Example: A bank database application consists of the customer data of the bank in addition to the programs that will implenet all the customer transactions such as deposit and withdrawal.

- Designing a Database application involves designing
  - The database
  - And the associated programs.

- Database Design Process

- The first two steps in designing a database are:
  - Collecting and analyzing requirements
  - Conceptual Design

-- Database Design Process



Miniworld

REQUIREMENTS COLLECTION AND ANALYSIS

Functional Requirements

Database 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

- Requirements

- The first step in designing a database is collecting, documenting, and analyzing requirements.
- In the next few slides we will show requirements of:
  - A Company Database
  - A University Database

-- Company Database Requirements ...

- The company is organized into departments.
- Each department:
  - Has a unique number
  - Has a unique name
  - Has a manager
  - May have several locations
- A department controls a number of projects.
- Each project:
  - Has a unique number
  - Has a unique name
  - Has one location
- The company stores each employee’s:
  - Social security number
  - Name
  - Address
  - Salary
  - Sex
  - Date of birth

… -- Company Database Requirements

- The company:
  - Assigns an employee to one department but may work on several projects, which are not necessarily controlled by the same department
  - keeps track of the number of hours per week that an employee works on each project.
  - Keeps track of the direct supervisor of each employee
  - Keeps track on the dependants of each employee for insurance purposes. It keeps each dependants:
    - Name
    - Sex
    - Birth date
    - Relationship to employee

-- University Database Requirements ...

- The university keeps track of each student
  - Unique number
  - Unique social security number
  - Name
  - current address
  - Current phone
  - Permanent address
    - City
    - State
    - Zip code
  - Permanent phone
  - Birth date
  - Sex
  - Class (freshman, sophomore, ...)
  - Department
  - Degree program (BA, BS, ..., PhD)
- The university also keeps track of each department
  - Name (unique)
  - Department code (unique)
  - Office number
  - Office phone
  - collage

-- University Database Requirements ...

- Each course has:
  - Name
  - Description
  - Number (unique)
  - Number of semester hours
  - Level
  - Offering department
- Each section has:
  - Number (unique per semester per course)
  - Semester
  - Year
  - Course
  - An instructor
- Each grade report has:
  - student number
  - section number
  - letter grade
  - Numeric grade (0, 1, 3, or 4)

- Conceptual Design

- Conceptual modeling +
- Entity-Relationship model +

-- Conceptual Modeling

- Once the requirements have being collected and analyzed, the next step is to create a conceptual schema for the database.
- A Conceptual schema is a concise description of the data requirements of the user and it contains:
  - Detailed description of the entity types
  - Relationships
  - constraints
- Conceptual schema is created using a high level conceptual data model
- Here, we will present the Entity-Relationship model, which is a popular high-level conceptual model, and we will use it in designing a database application.

- Entity Relationship Model

- Definition +
- Entities +
- Attributes +

-- Definition

- The Entity-Relationship (ER) model is a high-level conceptual data model that is widely used in the design of a database application.
- The ER model represents data in terms of:
  - **Entities**
  - **Attributes** of entities
  - **Relationships** between entities

-- Entities

- An entity is an object or a concept that is identified by the enterprise as having an independent existence.
- There are two types of entities:
  - Physical entities
    - Example: car, house, employee, part, supplier
  - Conceptual entities
    - Example: job, university course, sale, work experience

-- Entities of Company & University Databases

- Entities of:
  - Company database
    - Employees
    - Departments
    - Projects
    - Dependents
  - University Database
    - Students
    - Departments
    - Courses
    - Sections
    - Grade reports

-- Attributes

- Definition +
- Types of attributes +

--- Definition

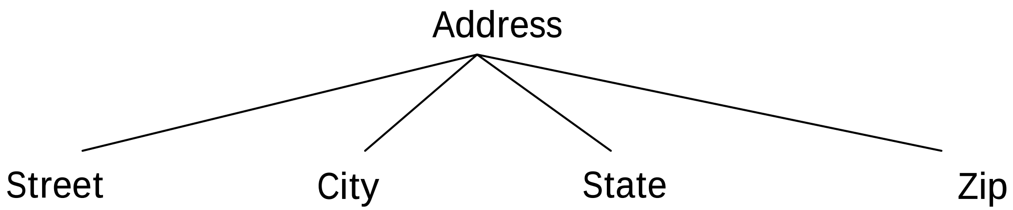
- An attribute is a property that describes an entity.
- Example:
  - The attributes of the entity CAR are:
    - Make
    - Chassis number
    - Color
  - The attributes of the entity EMPLOYEE are:
    - Name
    - Date of birth
    - Address
- There are many types of attributes.

--- Types of Attributes

- Simple Vs Composite Attributes +
- Single-valued Vs Multi-valued Attributes +
- Derived Vs Stored Attributes +
- Complex attributes +

---- Simple Vs Composite Attributes

- A simple attribute:
  - Consists of a single component with an independent existence.
    - Example: The **Sex** attribute of an EMPLOYEE entity
- A composite attribute:
  - Consists of multiple components each with an independent existence.
    - Example: The **Address** attribute of an EMPLOYEE entity can be divided into Street, City, State, Zip.



Address

Street City State Zip

---- Single-valued Vs Multi-valued Attributes

- Single-valued attribute:
  - Holds a single value for a single entity.
    - Example: the **Sex** attribute of an EMPLOYEE entity.

- Multi-valued attribute:
  - An attribute that holds more than one value for a single entity.
    - Example: The **Color** attribute of a CAR entity.

---- Derived Vs Stored Attributes

- Derived attribute:
  - Is an attribute that represents a value that is derived from the value of a related attribute or set of attributes not necessarily in the same entity.
    - Example 1: The value of the A**ge** attribute of the EMPLOYEE entity can be derived from the today's date and the value of the employee BirthDate.
    - Example 2: The **NumberofEmployees** attribute of a DEPARTMENT entity can be derived from the EMPLOYEE table by counting the number of employees who work in that department.

- Stored attribute:
  - Is an attribute that is not derived but which is directly stored in the entity.
    - Example: The S**ex** attribute in the EMPLOEE entity.

---- Complex attribute

- Is an attribute that is a nested combination of composite and multi-valued attributes in an arbitrary way.
  - Example: The complex attribute AddressPhone.

    {AdressPhone({Phone(AreaCode, PhoneNumber)},
    Address(StreetAddress(Number, Name, HouseNumber),
    City, State, Zip))}

  - Note:
    - Comma is used for separating the components
    - {} Represents multi-valued attributes
    - () is used for arbitrary nesting and grouping