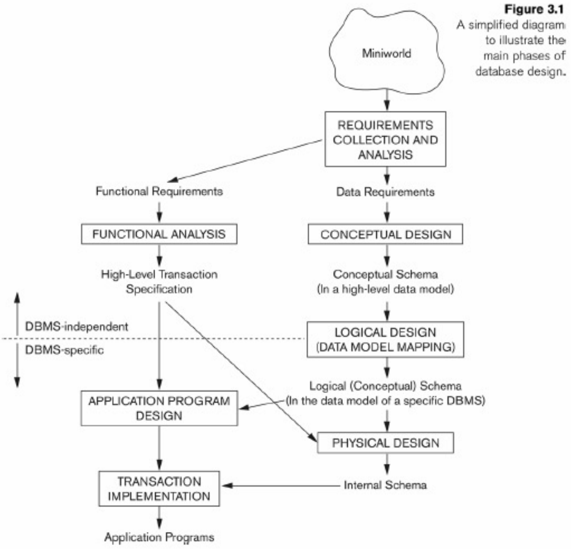
CSC 430/530 : DATABASE MANAGEMENT SYSTEMS/ DATABASE THEORY

Lecture 4: Introduction to database design primitives: The ER Model

Overview of Database Design Process



Miniworld

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

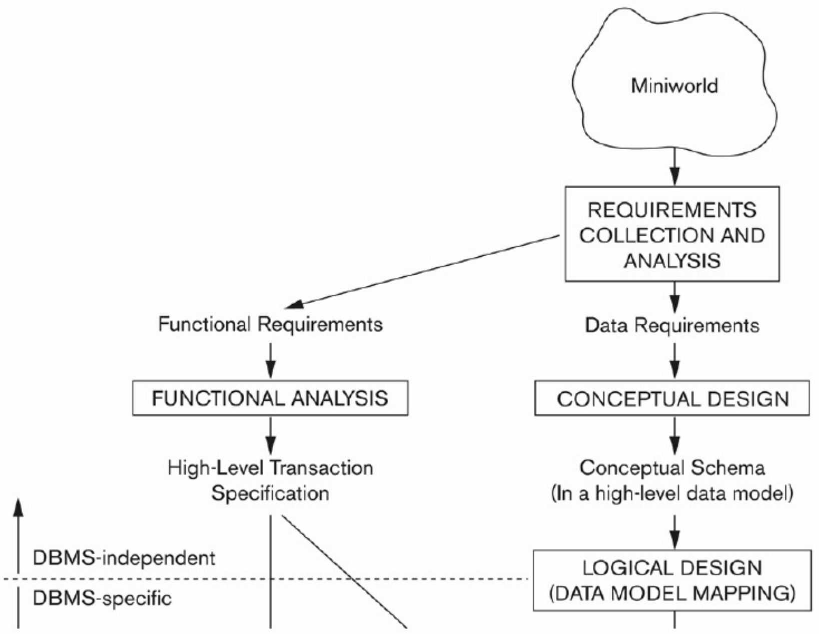
PHYSICAL DESIGN

TRANSACTION IMPLEMENTATION

Internal Schema

Application Programs

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



Miniworld

REQUIREMENTS COLLECTION AND ANALYSIS

Functional Requirements

Data Requirements

FUNCTIONAL ANALYSIS

CONCEPTUAL DESIGN

High-Level Transaction Specification

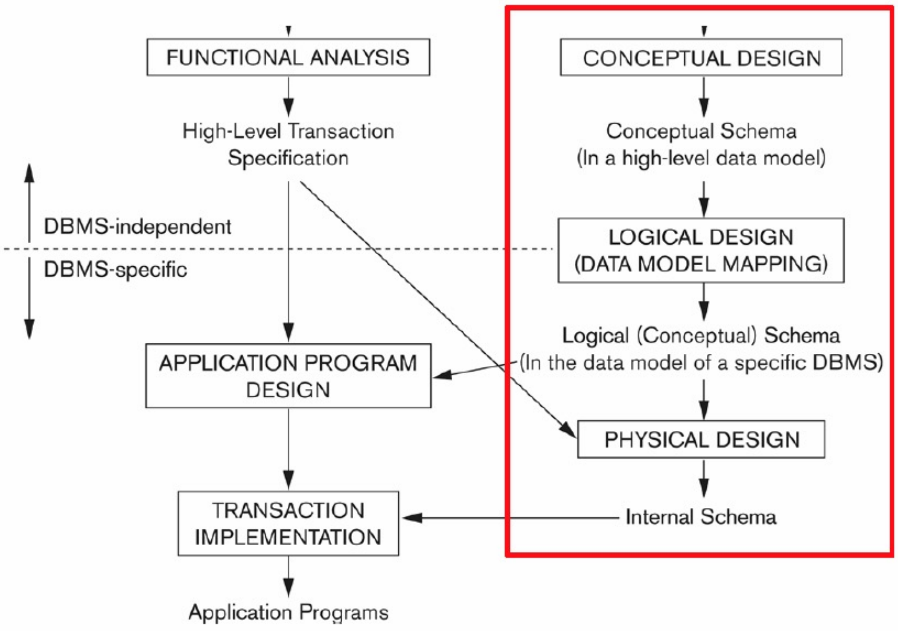
Conceptual Schema (In a high-level data model)

DBMS-independent

DBMS-specific

LOGICAL DESIGN (DATA MODEL MAPPING)

115



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

116

Questions

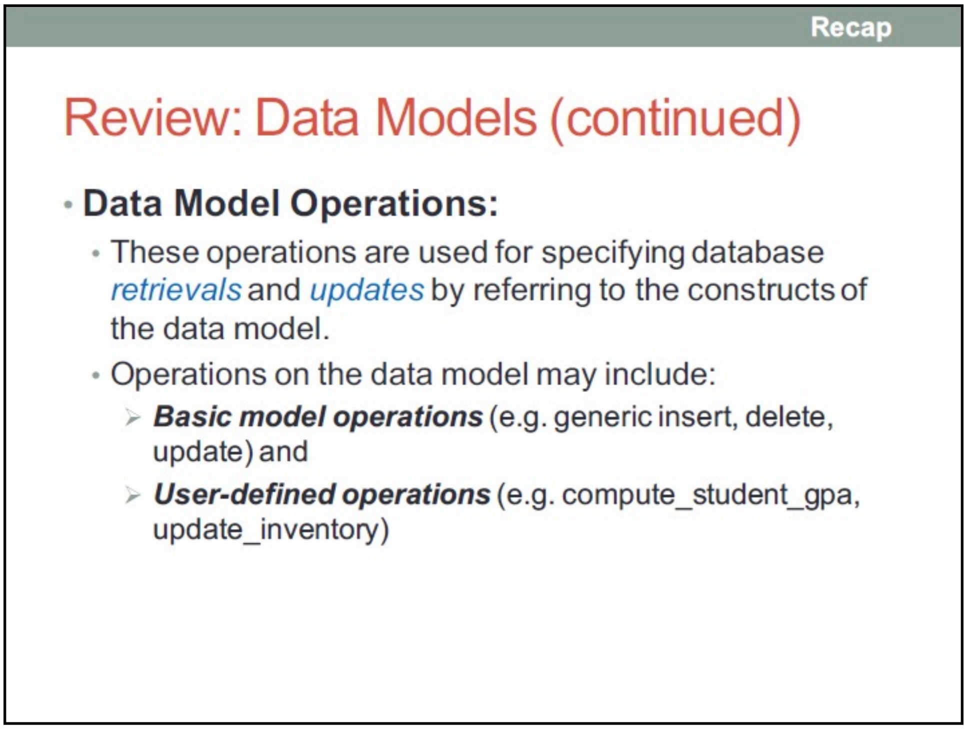
117

Recap

Review: Data Models

- **Data Model:**
  - A set of concepts to describe the ***structure*** of a database, the ***operations*** for manipulating these structures, and certain ***constraints*** that the database should obey.
- **Data Model Structure and Constraints:**
  - **Constructs** are used to define the database structure
  - **Constructs** typically include ***elements*** (and their ***data types***) as well as ***groups of elements*** (e.g. ***entity, record, table***), and ***relationships*** among such groups
  - ***Constraints*** specify some restrictions on valid data; these constraints must be enforced at all times

118

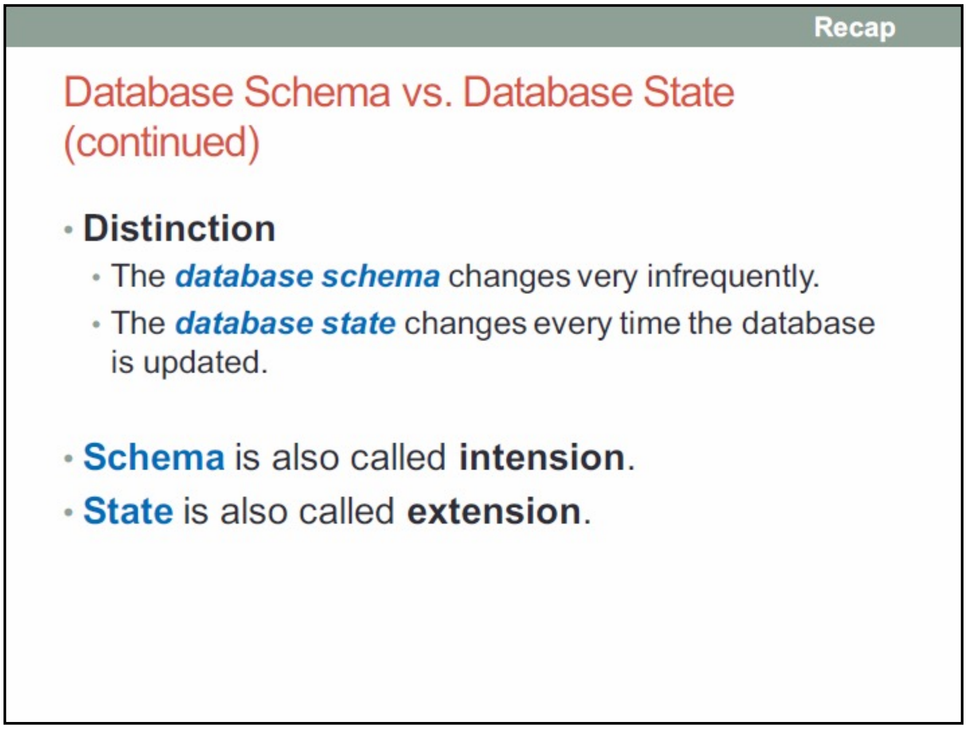


Recap

Review: Data Models (continued)

- **Data Model Operations:**
  - These operations are used for specifying database *retrievals* and *updates* by referring to the constructs of the data model.
  - Operations on the data model may include:
    - ➢ ***Basic model operations*** (e.g. generic insert, delete, update) and
    - ➢ ***User-defined operations*** (e.g. compute_student_gpa, update_inventory)

119



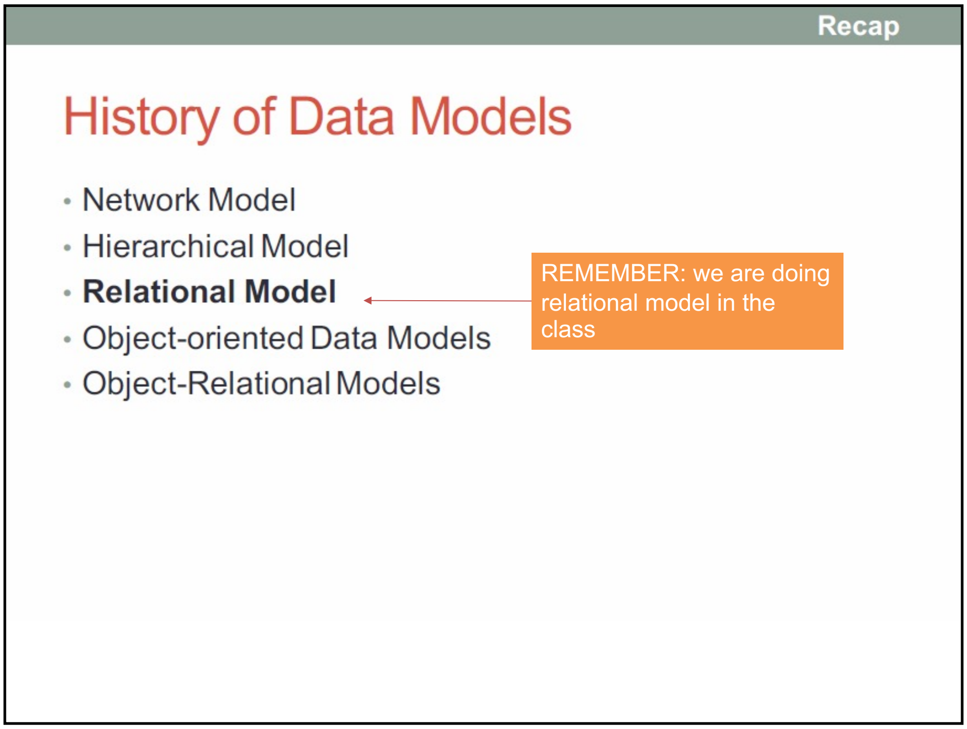
Recap

Database Schema vs. Database State (continued)

- **Distinction**
  - The ***database schema*** changes very infrequently.
  - The ***database state*** changes every time the database is updated.

- **Schema** is also called **intension**.
- **State** is also called **extension**.

120



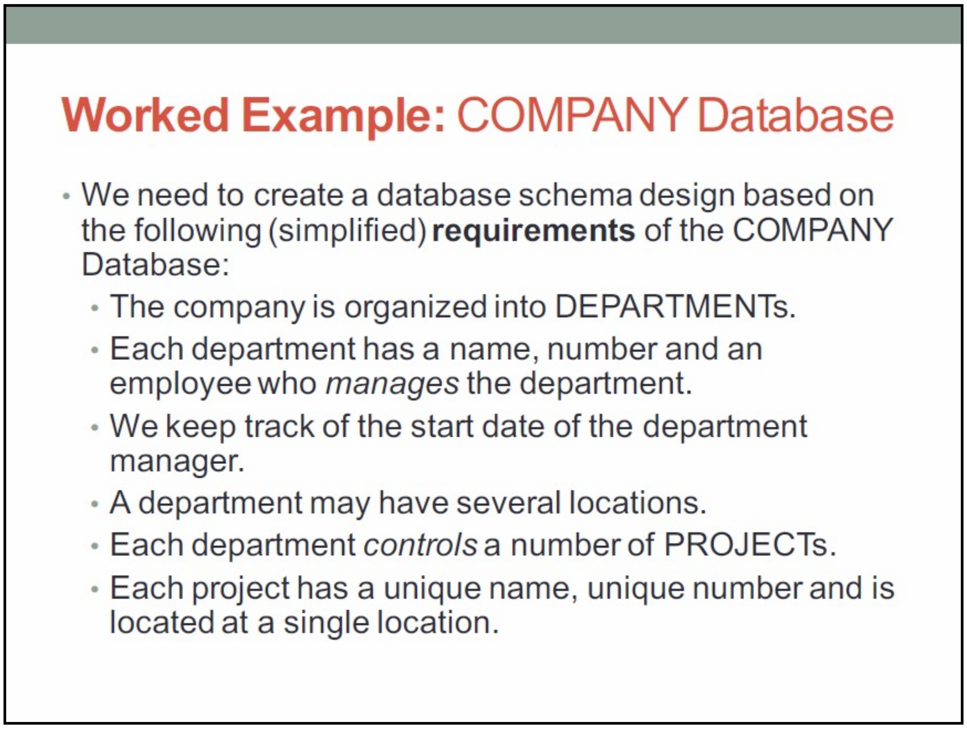
Recap

History of Data Models

- Network Model
- Hierarchical Model
- **Relational Model**
- Object-oriented Data Models
- Object-Relational Models

REMEMBER: we are doing relational model in the class

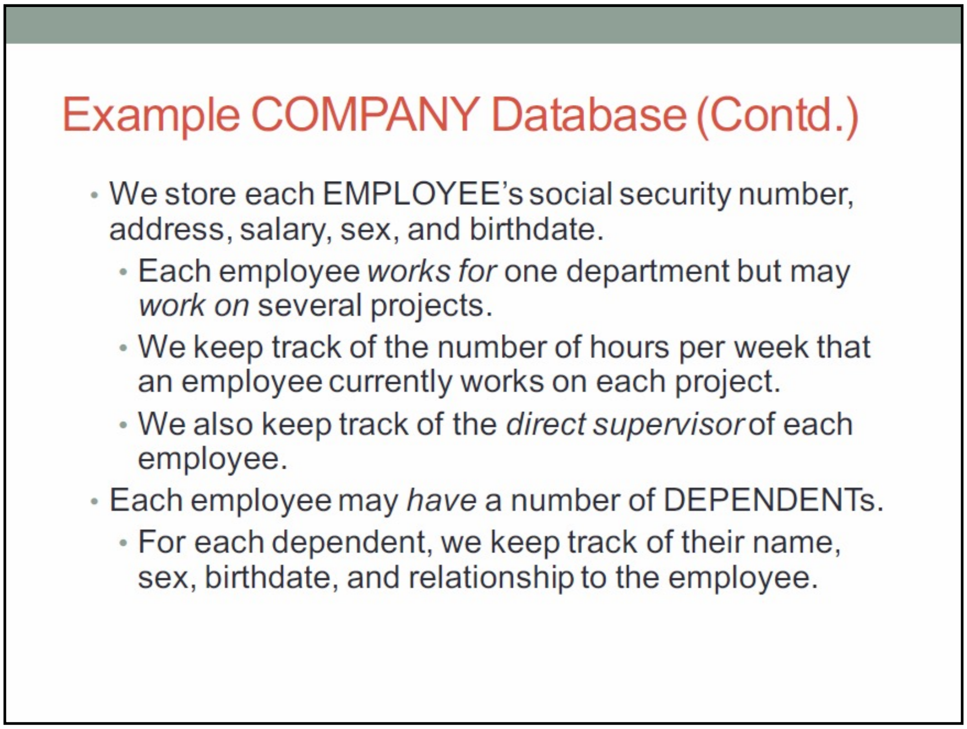
127



Worked Example: COMPANY Database

- We need to create a database schema design based on the following (simplified) **requirements** of the COMPANY Database:
  - The company is organized into DEPARTMENTs.
  - Each department has a name, number and an employee who *manages* the department.
  - We keep track of the start date of the department manager.
  - A department may have several locations.
  - Each department *controls* a number of PROJECTs.
  - Each project has a unique name, unique number and is located at a single location.

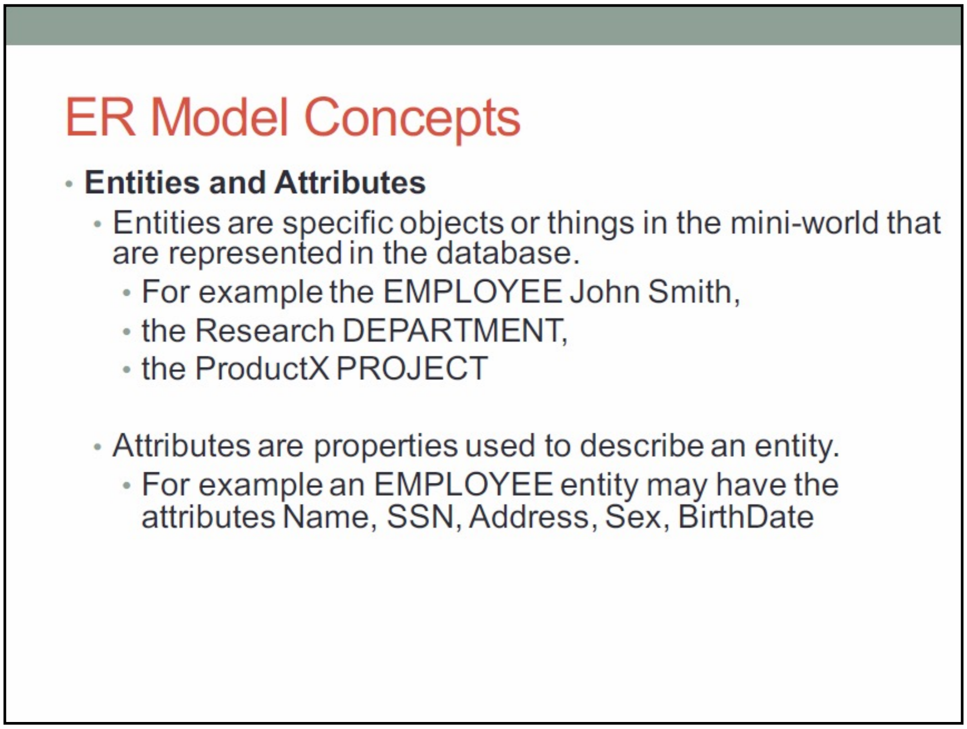
132



Example COMPANY Database (Contd.)

- We store each EMPLOYEE's social security number, address, salary, sex, and birthdate.
  - Each employee *works for* one department but may *work on* several projects.
  - We keep track of the number of hours per week that an employee currently works on each project.
  - We also keep track of the *direct supervisor* of each employee.
- Each employee may *have* a number of DEPENDENTs.
  - For each dependent, we keep track of their name, sex, birthdate, and relationship to the employee.

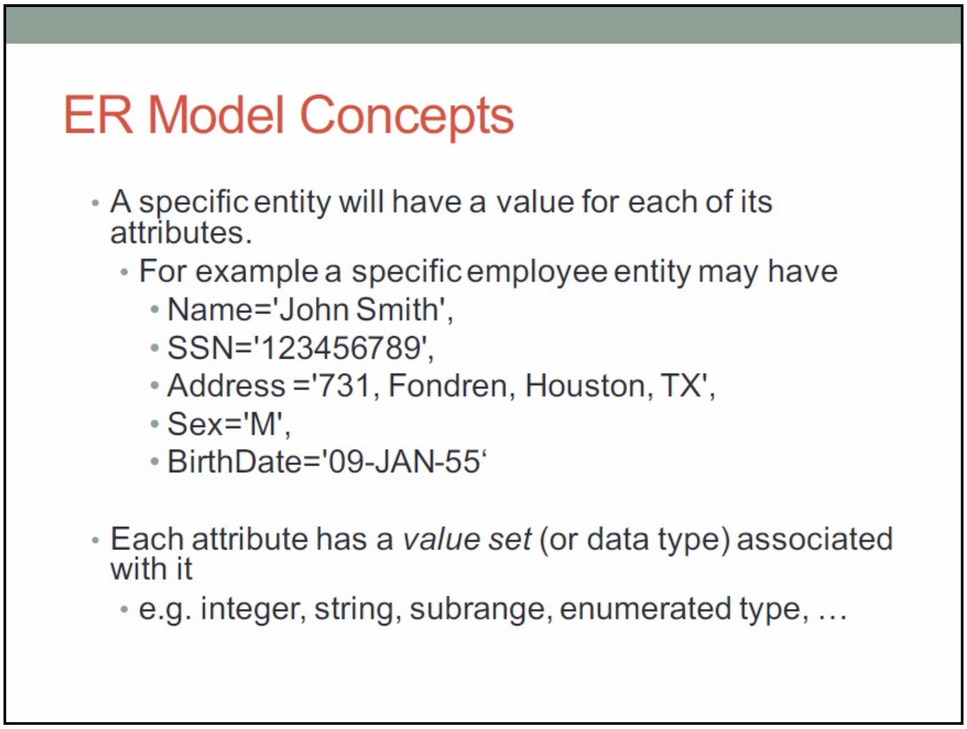
133



ER Model Concepts

- **Entities and Attributes**
  - Entities are specific objects or things in the mini-world that are represented in the database.
    - For example the EMPLOYEE John Smith,
    - the Research DEPARTMENT,
    - the ProductX PROJECT
  - Attributes are properties used to describe an entity.
    - For example an EMPLOYEE entity may have the attributes Name, SSN, Address, Sex, BirthDate

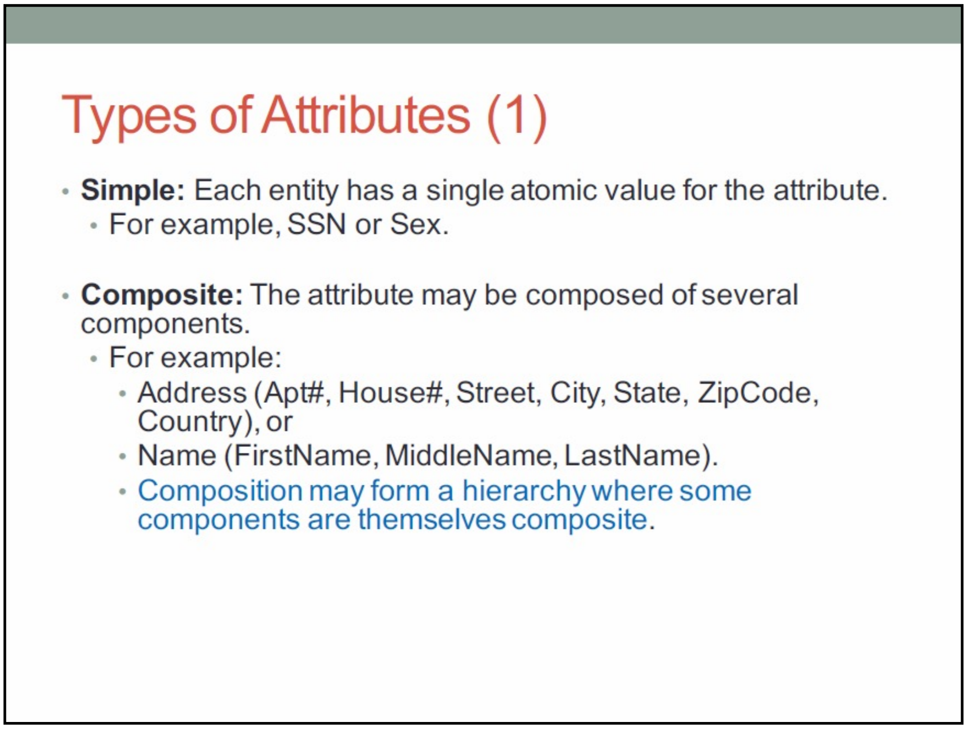
134



ER Model Concepts

- A specific entity will have a value for each of its attributes.
  - For example a specific employee entity may have
    - Name='John Smith',
    - SSN='123456789',
    - Address ='731, Fondren, Houston, TX',
    - Sex='M',
    - BirthDate='09-JAN-55‘
- Each attribute has a *value set* (or data type) associated with it
  - e.g. integer, string, subrange, enumerated type, …

135



Types of Attributes (1)

- **Simple:** Each entity has a single atomic value for the attribute.
  - For example, SSN or Sex.
- **Composite:** The attribute may be composed of several components.
  - For example:
    - Address (Apt#, House#, Street, City, State, ZipCode, Country), or
    - Name (FirstName, MiddleName, LastName).
    - Composition may form a hierarchy where some components are themselves composite.

136

Example of a composite attribute

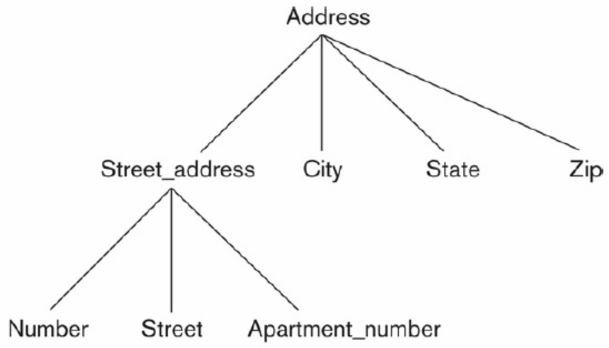


Figure 3.4
A hierarchy of composite attributes.

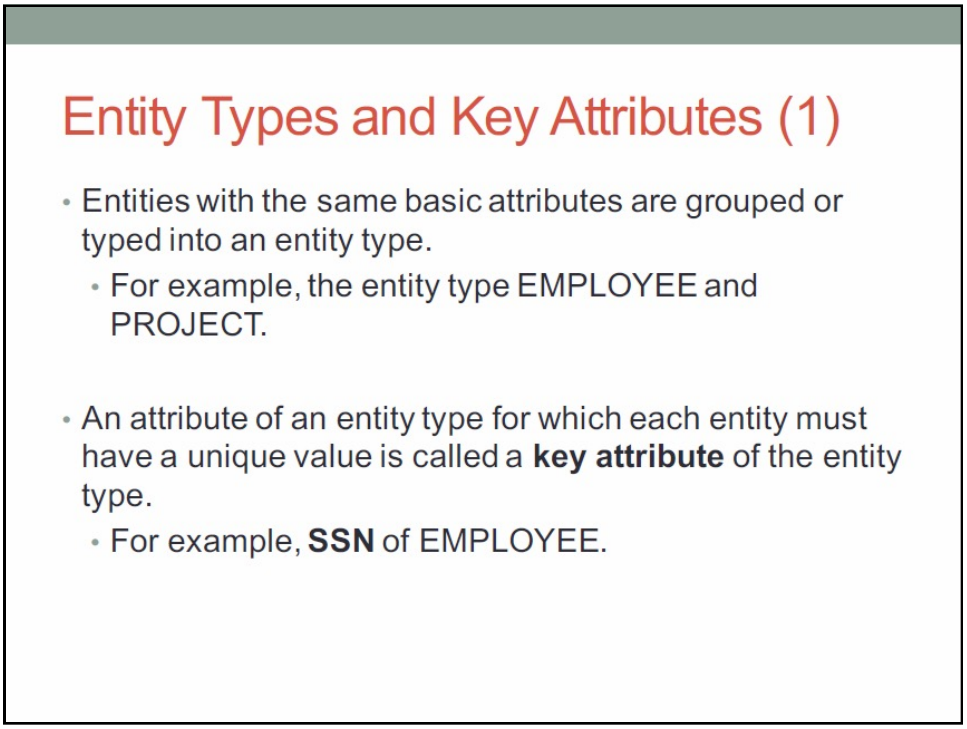
137

Types of Attributes (2)

- **Multi-valued:** An entity may have multiple values for that attribute.
  - For example, PreviousDegrees of a STUDENT.
    - Denoted as {PreviousDegrees}.

- In general, composite and multi-valued attributes may be nested arbitrarily to any number of levels, although this is rare.
  - For example, PreviousDegrees of a STUDENT is a composite multi-valued attribute denoted by {PreviousDegrees (College, Year, Degree, Field)}
    - Multiple PreviousDegrees values can exist
    - Each has four subcomponent attributes:
      - College, Year, Degree, Field

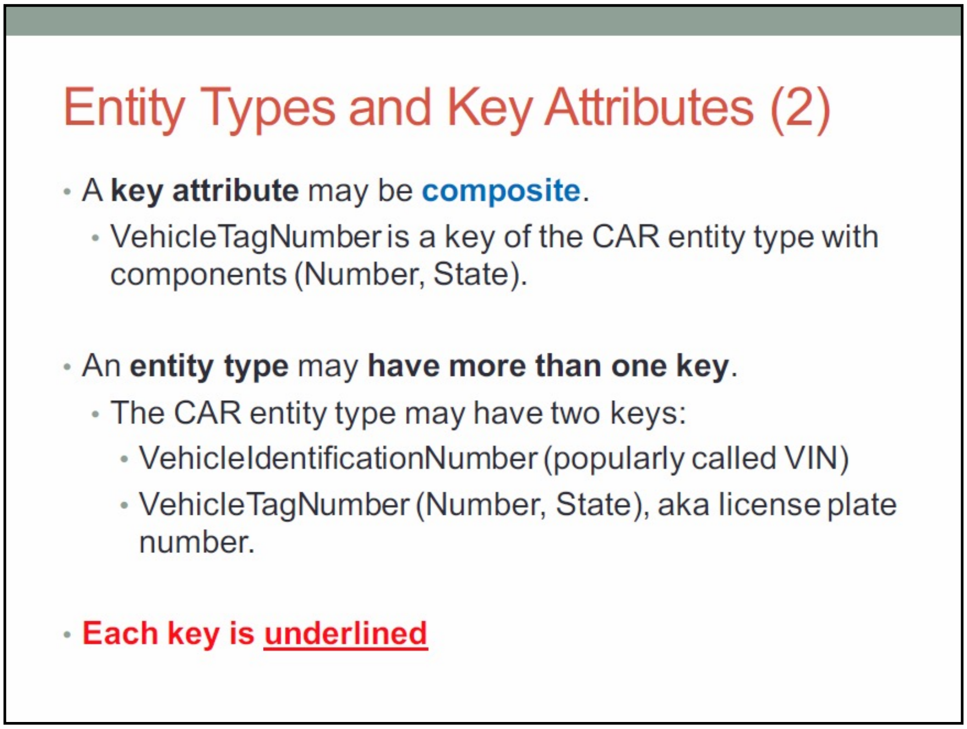
138



Entity Types and Key Attributes (1)

- Entities with the same basic attributes are grouped or typed into an entity type.
  - For example, the entity type EMPLOYEE and PROJECT.
- An attribute of an entity type for which each entity must have a unique value is called a **key attribute** of the entity type.
  - For example, **SSN** of EMPLOYEE.

139



Entity Types and Key Attributes (2)

- A **key attribute** may be **composite**.
  - VehicleTagNumber is a key of the CAR entity type with components (Number, State).
- An **entity type** may **have more than one key**.
  - The CAR entity type may have two keys:
    - VehicleIdentificationNumber (popularly called VIN)
    - VehicleTagNumber (Number, State), aka license plate number.
- **Each key is underlined**

140

Displaying an Entity type

- In ER diagrams, an **entity type** is displayed in a **rectangular box**

- **Attributes** are displayed in **ovals**
  - Each **attribute** is connected to its entity type
  - **Components of a composite attribute** are **connected to the oval representing the composite attribute**
  - Each key attribute is underlined
  - **Multivalued attributes** displayed in **double ovals**

- See CAR example on next slide

Entity Type CAR with two keys and a corresponding Entity Set

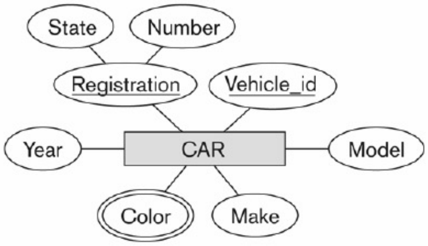
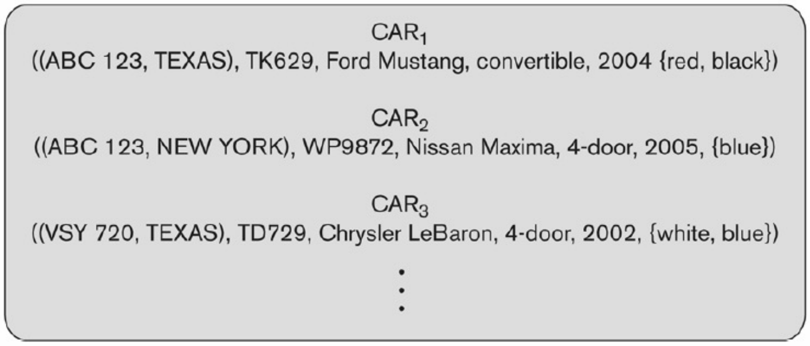


Figure 3.7
The CAR entity type with two key attributes, Registration and Vehicle_id. (a) ER diagram notation. (b) Entity set with three entities.

Entity Type CAR with two keys and a corresponding Entity Set

CAR
Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}



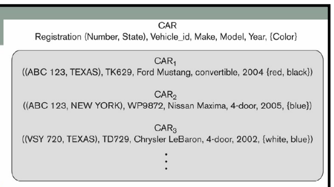
CAR$_1$
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

CAR$_2$
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

CAR$_3$
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮

143



CAR
Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}

CAR$_1$
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

CAR$_2$
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

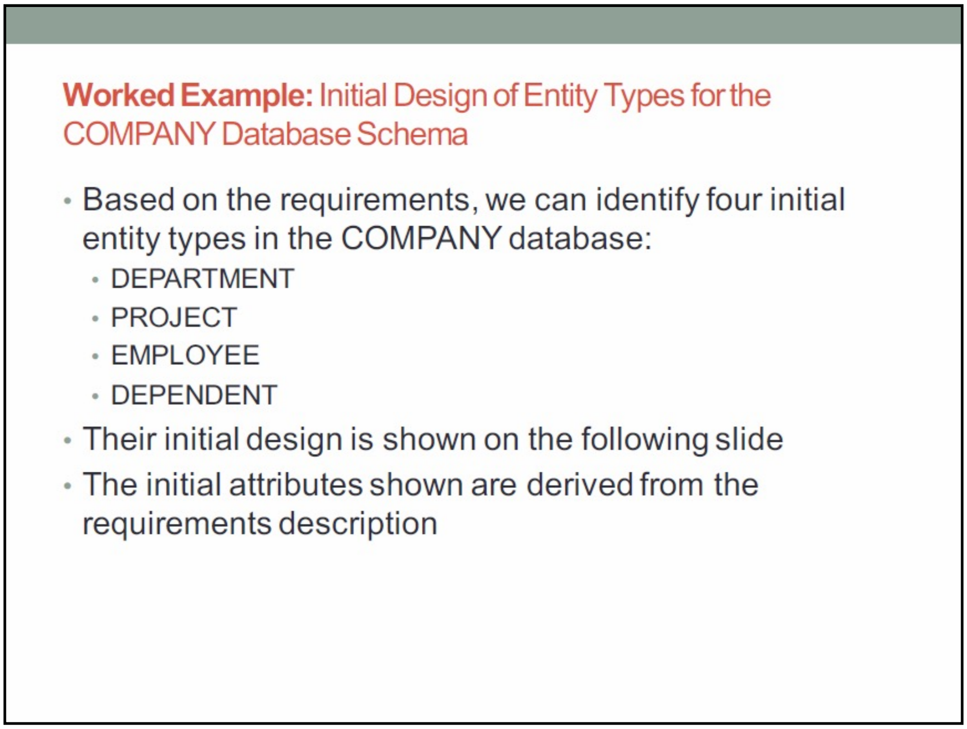
CAR$_3$
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮

Entity Set

- Each entity type will have a collection of entities stored in the database
  - Called the **entity set**
- Previous slide shows three CAR entity instances in the entity set for CAR
- Same name (CAR) used to refer to both the entity type and the entity set
- Entity set is the current *state* of the entities of that type that are stored in the database

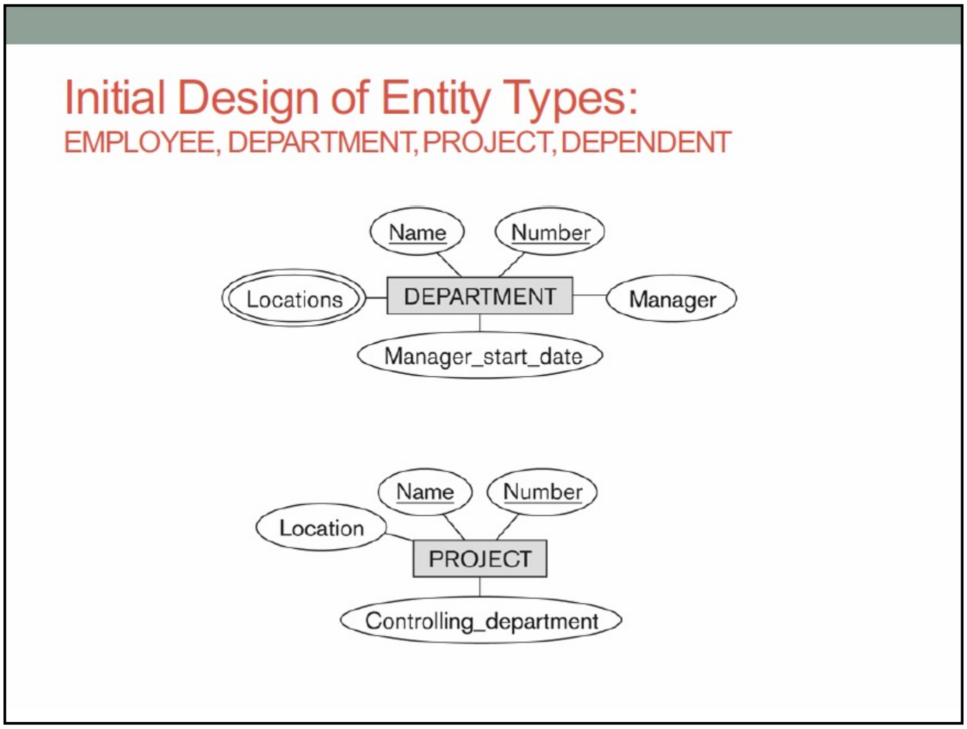
144



Worked Example: Initial Design of Entity Types for the COMPANY Database Schema

- Based on the requirements, we can identify four initial entity types in the COMPANY database:
  - DEPARTMENT
  - PROJECT
  - EMPLOYEE
  - DEPENDENT
- Their initial design is shown on the following slide
- The initial attributes shown are derived from the requirements description

145



Initial Design of Entity Types:
EMPLOYEE, DEPARTMENT, PROJECT, DEPENDENT

146

Initial Design of Entity Types:

EMPLOYEE, DEPARTMENT, PROJECT, DEPENDENT

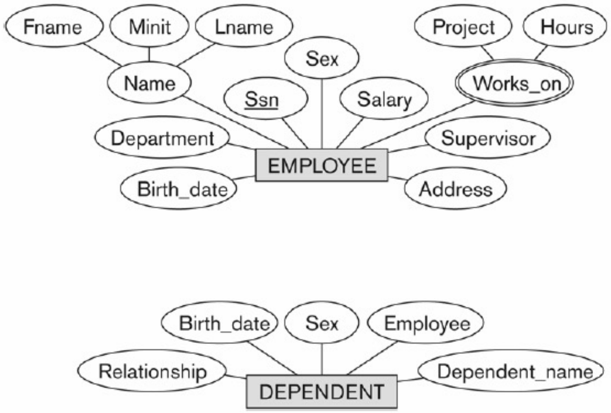


Figure 3.8
Preliminary design of entity types for the COMPANY database. Some of the shown attributes will be refined into relationships.

Accomplishment!!!

- **Congratulations!!!** we have our initial design.

- **Points to introspect**

1. Given the entity description and their corresponding Entity diagrams, can store data for multiple instances.
2. Can I say that I have an effective database?
3. What are the caveats to such a design?