Unit 2: Database Models

(Part 1)

Learning Objectives (1 of 2)

- In this chapter, you will learn:
  - About data modeling and why data models are important
  - About the basic data-modeling building blocks
  - What business rules are and how they influence database design

Learning Objectives (2 of 2)

- In this chapter, you will learn:
  - How the major data models evolved
  - About emerging alternative data models and the need they fulfill
  - ~~How data models can be classified by their level of abstraction~~

Outline

- Data Models – Building Blocks
- Business Rules
- Evolution of Data Models

Data Modeling and Data Models

- **Data modeling**: Iterative and progressive process of creating a specific data model for a determined problem domain
- **Data models**: Simple representations of complex real-world data structures
  - Useful for supporting a specific problem domain
- **Model** - Abstraction of a real-world object or event

Importance of Data Models

- Are a communication tool
- Give an overall view of the database
- Organize data for various users
- Are an abstraction for the creation of good database

Data Model Basic Building Blocks

- **Entity**: Unique and distinct object used to collect and store data
  - **Attribute**: Characteristic of an entity
- **Relationship**: Describes an association among entities
  - **One-to-many (1:M)**
  - **Many-to-many (M:N or M:M)**
  - **One-to-one (1:1)**
- **Constraint**: Set of rules to ensure data integrity

Business Rules

Business Rules

- Brief, precise, and unambiguous description of a policy, procedure, or principle
- Enable defining the basic building blocks
- Describe main and distinguishing characteristics of the data

Sources of Business Rules

- Company managers
- Policy makers
- Department managers
- Written documentation
- Direct interviews with end users

Reasons for Identifying and Documenting Business Rules

- Help standardize company’s view of data
- Communications tool between users and designers
- Allow designer to:
  - Understand the nature, role, scope of data, and business processes
  - Develop appropriate relationship participation rules and constraints
  - Create an accurate data model

Translating Business Rules into Data Model Components

- Nouns translate into entities
- Verbs translate into relationships among entities
- Relationships are bidirectional
- Questions to identify the relationship type
    - How many instances of B are related to one instance of A?
    - How many instances of A are related to one instance of B?

Exercise: Consider the Business Rule

"A customer may generate many invoices"

- How many entities? (nouns)
- How many relationships (verbs)

Exercise: Consider the Business Rule

"A customer may generate many invoices"

- *Customer* and *invoice* are objects of interest (Entities)
- There is a *generate* relationship

Exercise: Consider the Business Rules

"A customer may generate many invoices"

"an invoice is generated by only one customer."

- What is the relationship between both entities?
  - How many instances of B are related to one instance of A?
  - How many instances of A are related to one instance of B?

Exercise: What is the relationship?

- What is the relationship between a student and a course in a university environment?

  - How many instances of B are related to one instance of A?
  - How many instances of A are related to one instance of B?

Exercise: What is the relationship?

- What is the relationship between a student and a course in a university environment
  - How many instances of B are related to one instance of A?
  - How many instances of A are related to one instance of B?

- In how many classes can one student enroll? Answer: many classes.
- How many students can enroll in one class? Answer: many students.
- Therefore we have a many-to-many relationship.

Naming Conventions

- Entity names - Required to:
  - Be descriptive of the objects in the business environment
  - Use terminology that is familiar to the users
- Attribute name - Required to be descriptive of the data represented by the attribute
- Proper naming:
  - Facilitates communication between parties
  - Promotes self-documentation

CENGAGE

Evolution of Data Models

Hierarchical and Network Models

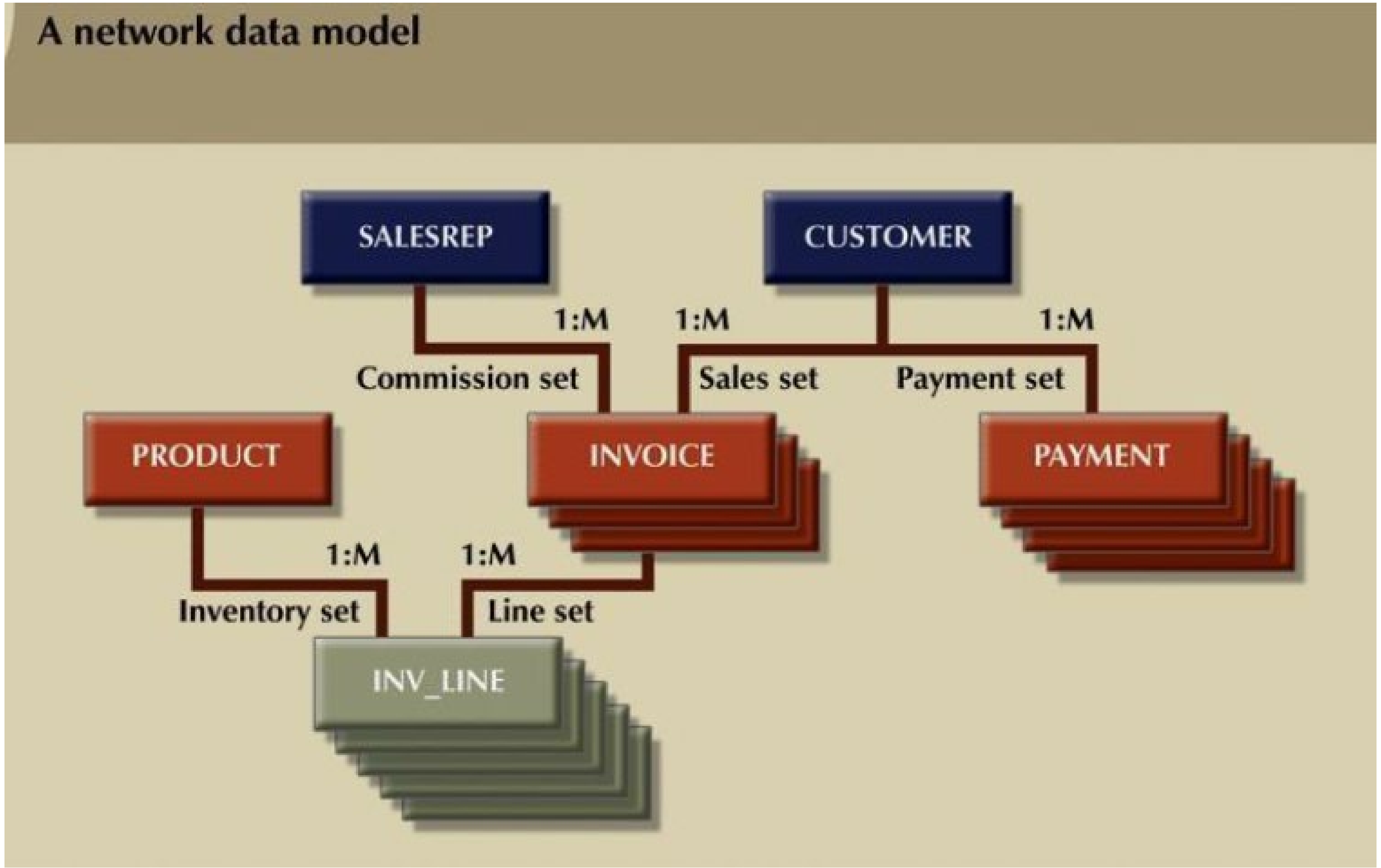
Hierarchical Models

- Developed to manage large amounts of data for complex manufacturing projects
- Represented by an upside-down tree which contains **segments** (equivalent of a file system's record type)
- Depicts a set of one-to-many (1:M) relationships

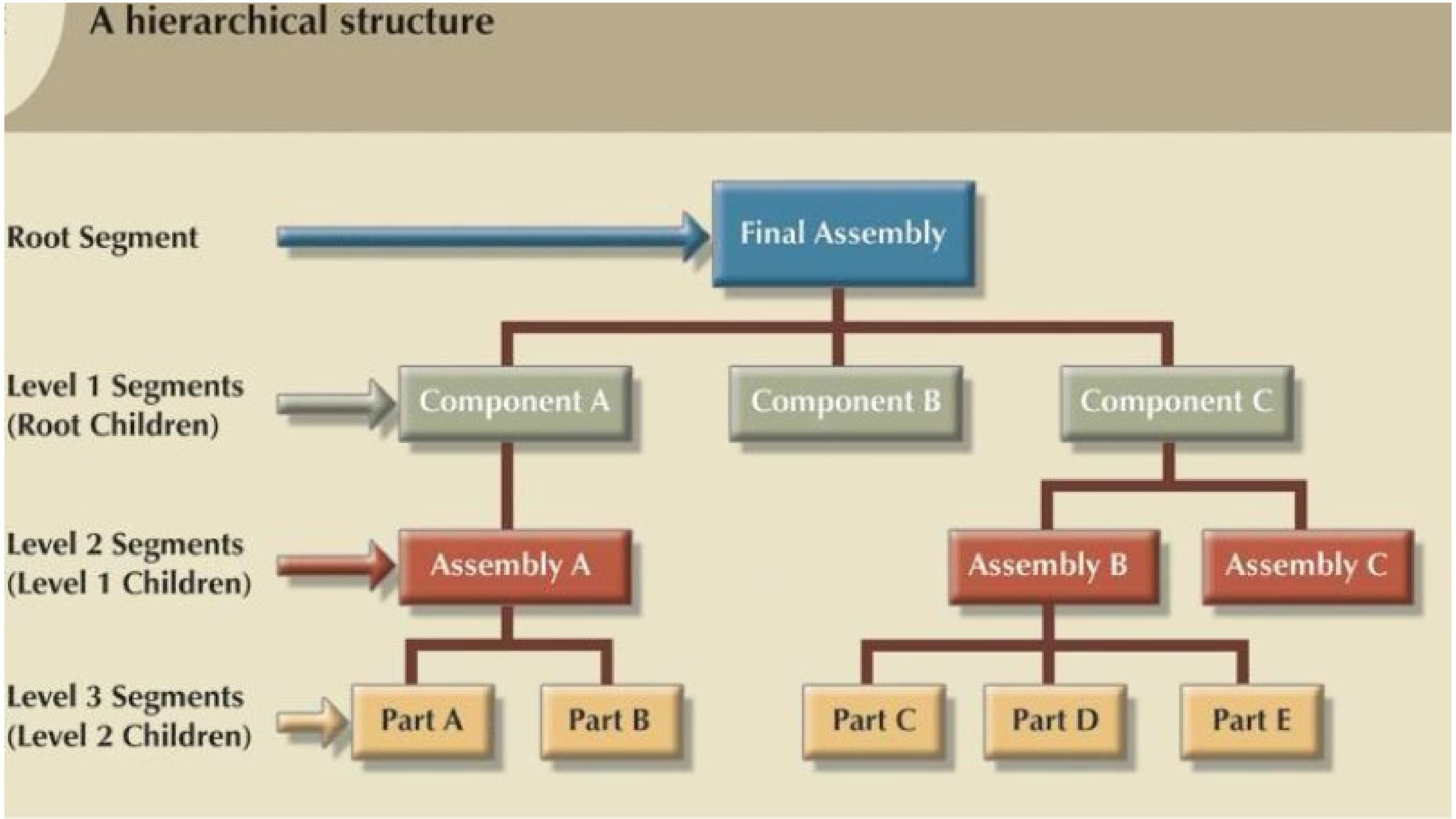
Network Models

- Created to represent complex data relationships effectively
- Improved database performance and imposed a database standard
- Allows a record to have more than one parent
- Depicts both one-to-many (1:M) and many-to-many (M:N) relationships

A Network Data Model



A Hierarchical Data Model



Standard Database Concepts (1 of 2)

- **Schema**
  - Conceptual organization of the entire database as viewed by the database administrator
- **Subschema**
  - Portion of the database seen by the application programs that produce the desired information from the data within the database

Standard Database Concepts (2 of 2)

- **Data manipulation language (DML)**
  - Environment in which data can be managed and is used to work with the data in the database
- **Schema data definition language (DDL)**
  - Enables the database administrator to define the schema components

The Relational Model

- Based on a relation
  - **Relation** or **table**: Matrix composed of intersecting tuple and attribute
    - **Tuple**: Rows
    - **Attribute**: Columns
- Describes a precise set of data manipulation constructs

The Relational Model - 2

- Tables store collection of related entities
- Tables are related to each other through the sharing of a common attribute
- Relational diagram
  - Representation of entities, attributes, and relationships

FIGURE 2.1 LINKING RELATIONAL TABLES

Table name: AGENT (first six attributes)

Database name: Ch02_InsureCo

AGENT_CODE	AGENT_LNAME	AGENT_FNAME	AGENT_INITIAL	AGENT_AREACODE	AGENT_PHONE
501	Alby	Alex	B	713	228-1249
502	Hahn	Leah	F	615	882-1244
503	Okon	John	T	615	123-5589

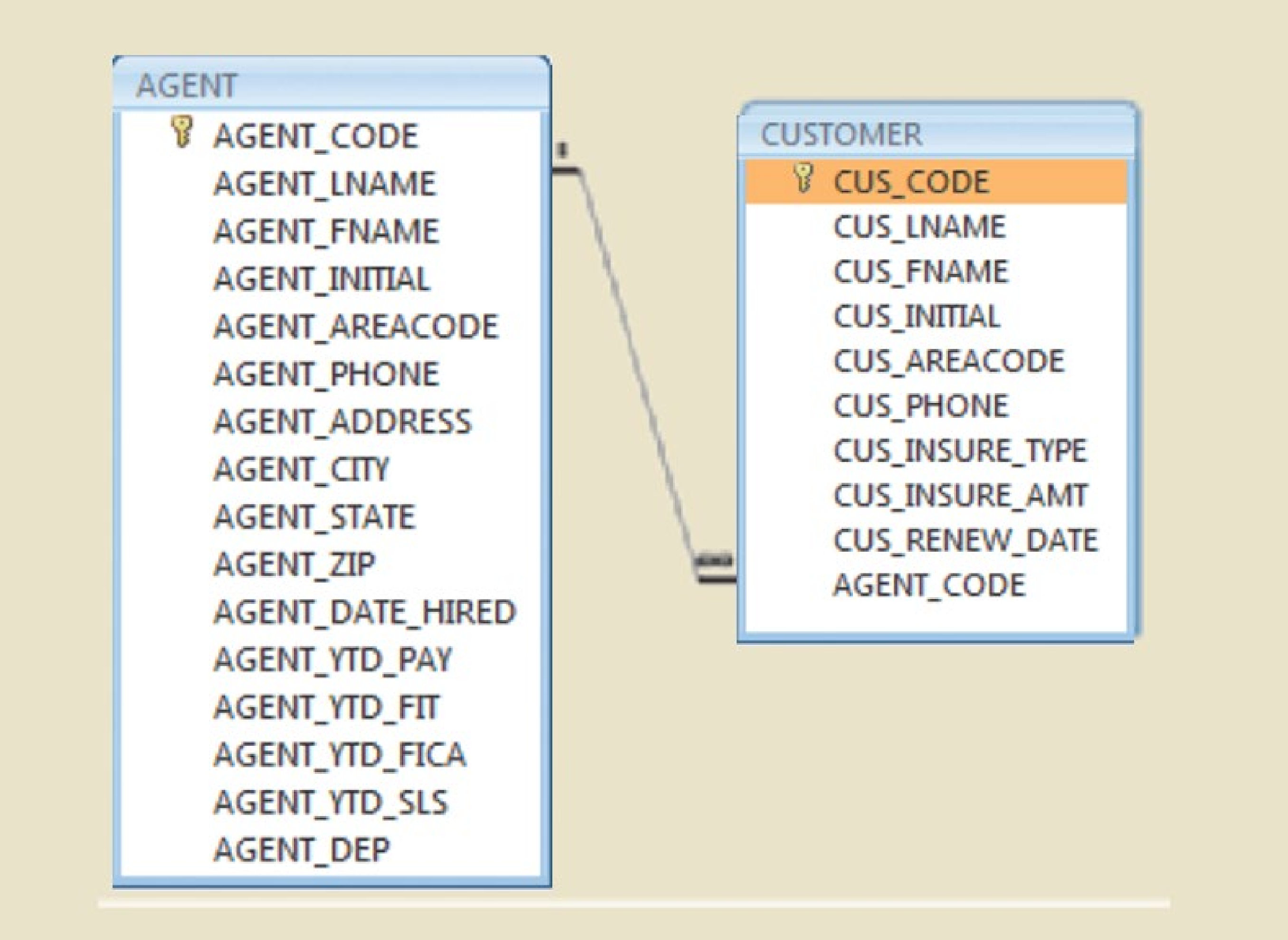
Link through AGENT_CODE

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_INSURE_TYPE	CUS_INSURE_AMT	CUS_RENEW_DATE	AGENT_CODE
10010	Ramas	Alfred	A	615	844-2573	T1	100.00	05-Apr-2016	502
10011	Dunne	Leona	K	713	894-1238	T1	250.00	16-Jun-2016	501
10012	Smith	Kathy	W	615	894-2285	S2	150.00	29-Jan-2017	502
10013	Olowski	Paul	F	615	894-2180	S1	300.00	14-Oct-2016	502
10014	Orlando	Myron		615	222-1672	T1	100.00	28-Dec-2017	501
10015	O'Brian	Amy	B	713	442-3381	T2	850.00	22-Sep-2016	503
10016	Brown	James	G	615	297-1228	S1	120.00	25-Mar-2017	502
10017	Williams	George		615	290-2556	S1	250.00	17-Jul-2016	503
10018	Farriss	Anne	G	713	382-7185	T2	100.00	03-Dec-2016	501
10019	Smith	Olette	K	615	297-3809	S2	500.00	14-Mar-2017	503

The relational model provides a minimum level of controlled redundancy to eliminate most of the redundancies commonly found in file systems.

Figure 2.2 - A Relational Diagram



Relational Database Management System (RDBMS)

- Performs basic functions provided by the hierarchical and network DBMS systems
- Makes the relational data model easier to understand and implement
- Hides the complexities of the relational model from the user

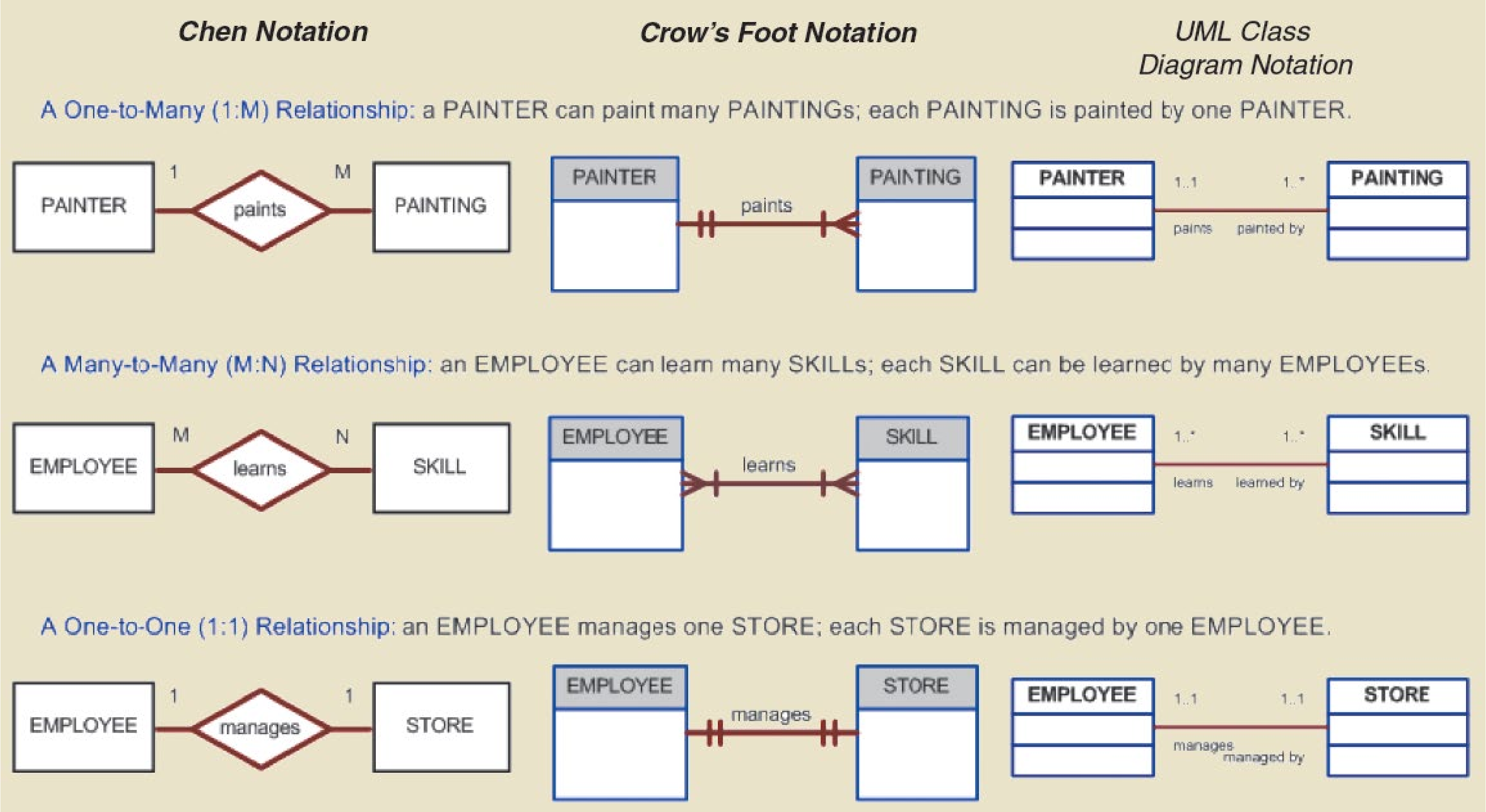
SQL-Based Relational Database Application

- End-user interface
  - Allows end user to interact with the data
- Collection of tables stored in the database
  - Each table is independent from another
  - Rows in different tables are related based on common values in common attributes
- SQL engine
  - Executes all queries

The Entity Relationship Model

- Graphical representation of entities and their relationships in a database structure
- **Entity relationship diagram (ERD)**
  - Uses graphic representations to model database components
- **Entity instance or entity occurrence**
  - Rows in the relational table
- **Connectivity**: Term used to label the relationship types

Figure 2.3 - The ER Model Notations



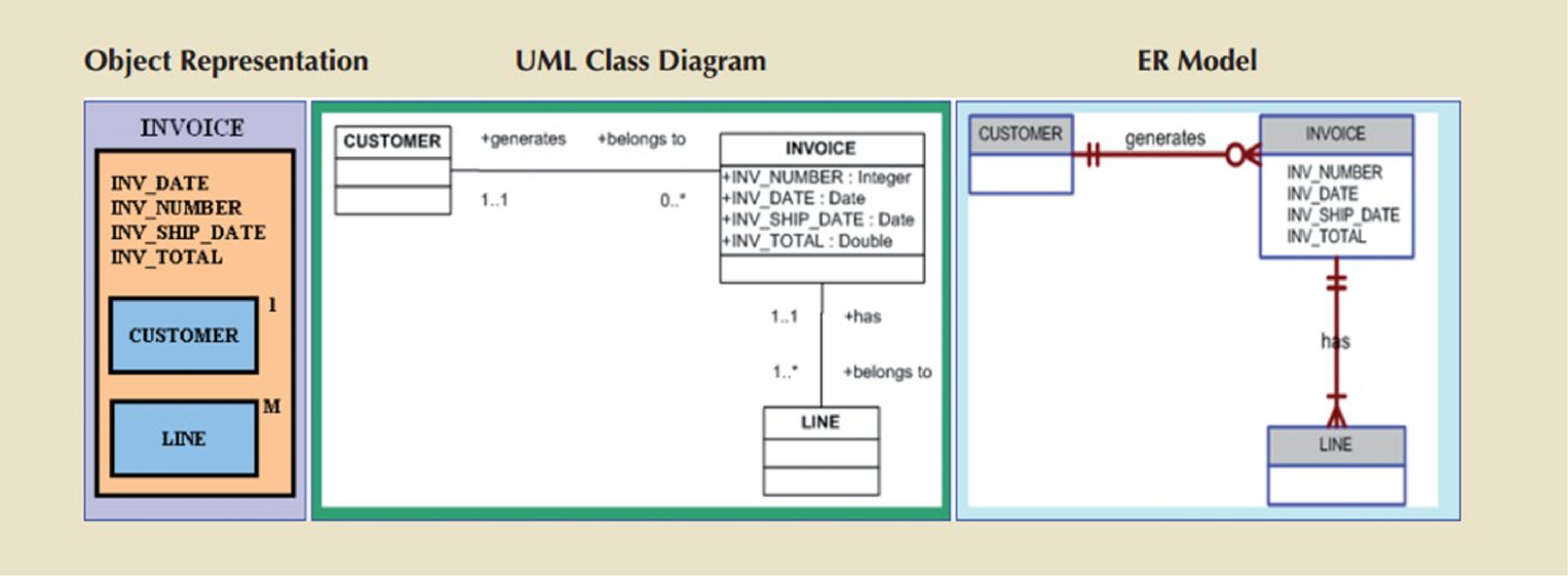
The Object-Oriented Data Model (OODM) or Semantic Data Model

- **Object-oriented database management system (OODBMS)**
  - Based on OODM
- **Object**: Contains data and their relationships with operations that are performed on it
  - Basic building block for autonomous structures
  - Abstraction of real-world entity
- Attributes - Describe the properties of an object

The Object-Oriented Data Model (OODM)

- **Class**: Collection of similar objects with shared structure and behavior organized in a class hierarchy
  - **Class hierarchy**: Resembles an upside-down tree in which each class has only one parent
- **Inheritance**: Object inherits methods and attributes of parent class
- **Unified Modeling Language (UML)**
  - Describes sets of diagrams and symbols to graphically model a system

Figure 2.4 - A Comparison of OO, UML and ER Models



Object/Relational and XML

- **Extended relational data model (ERDM)**
  - Supports OO features and complex data representation
  - **Object/Relational Database Management System (O/R DBMS)**
    - Based on ERDM, focuses on better data management
- **Extensible Markup Language (XML)**
  - Manages unstructured data for efficient and effective exchange of all data types

Big Data

- Aims to:
  - Find new and better ways to manage large amounts of web and sensor-generated data
  - Provide high performance and scalability at a reasonable cost
- Characteristics
  - Volume
  - Velocity
  - Variety

Big Data Challenges

- Volume does not allow the usage of conventional structures
- Expensive
- OLAP tools proved inconsistent dealing with unstructured data

Big Data New Technologies

- Hadoop
  - Hadoop Distributed File System (HDFS)
  - MapReduce
- NoSQL

NoSQL Databases

- Not based on the relational model
- Support distributed database architectures
- Provide high scalability, high availability, and fault tolerance
- Support large amounts of sparse data
- Geared toward performance rather than transaction consistency
- Store data in key-value stores

Figure 2.5 - A Simple Key-Value Representation (1 of 2)

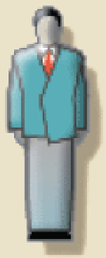
Trucks-R-Us

Data stored using traditional relational model

DID	CERT1	CERT2	CERT3	DOB	LICTYPE
2732	80		95	1/24/1962	P
2946		92		4/11/1970	
3650	86			11/27/1963	R

Data stored using key-value model

DID	KEY	VALUE
2732	CERT1	80
2732	CERT3	95
2732	DOB	1/24/1962
2732	LICTYPE	P
2946	CERT2	92
2946	DOB	4/11/1970
3650	CERT1	86
3650	DOB	11/27/63
3650	LICTYPE	R

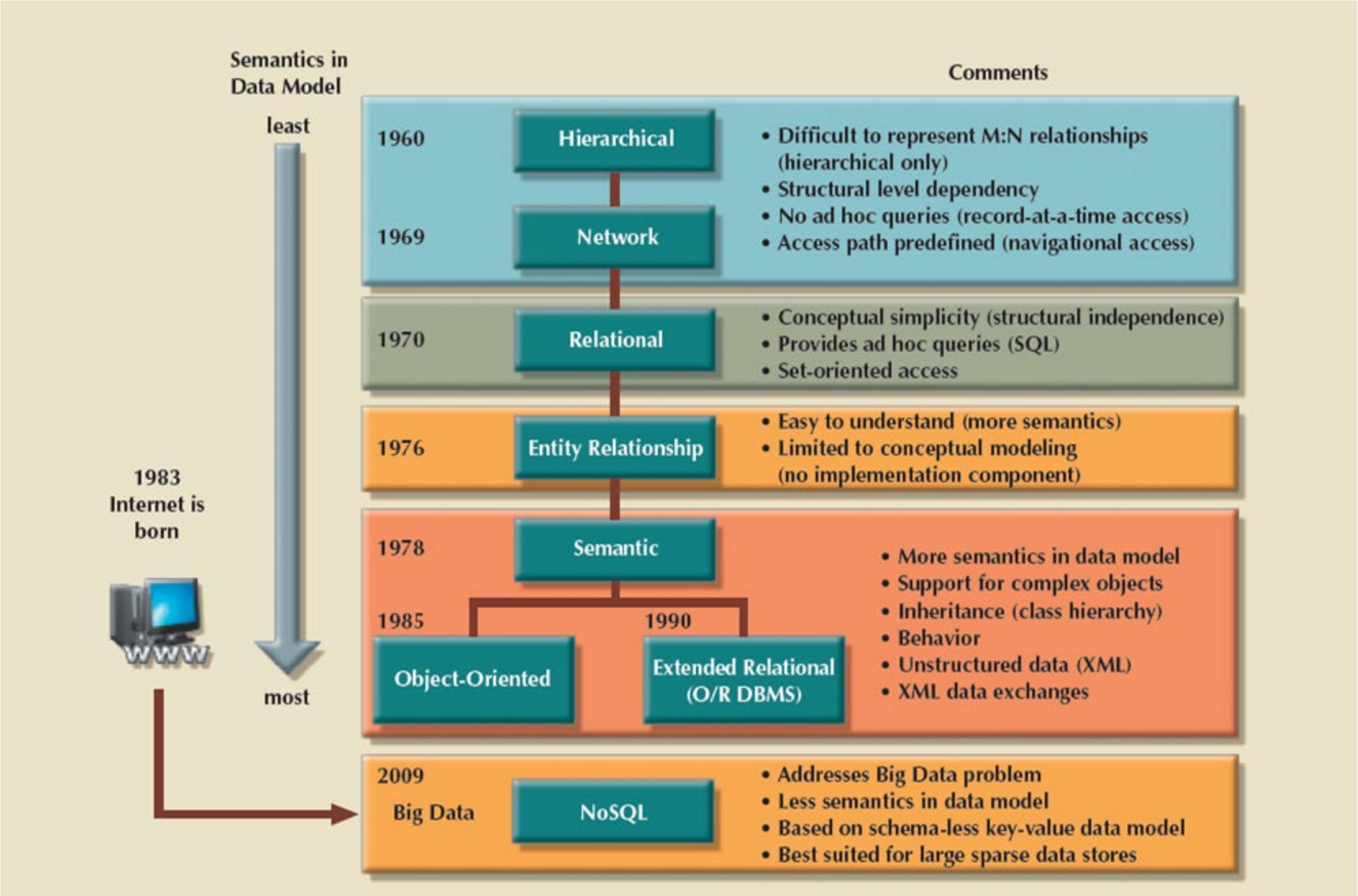


Driver 2732

Figure 2.5 - A Simple Key-Value Representation (2 of 2)

- In the relational model:
  - Each row represents one entity instance.
  - Each column represents one attribute of the entity.
  - The values in a column are of the same data type.
- In the key-value model:
  - Each row represents one attribute/value of one entity instance.
  - The “key” column could represent any entity’s attribute.
  - The values in the “value” column could be of any data type and therefore it is generally assigned a long string data type.

Figure 2.6 - The Evolution of Data Models



CENGAGE

Hierarchical Model

Advantages

- Promotes data sharing
- Parent/child relationship promotes conceptual simplicity and data integrity
- Database security is provided and enforced by DBMS
- Efficient with 1:M relationships

Disadvantages

- Requires knowledge of physical data storage characteristics
- Navigational system requires knowledge of hierarchical path
- Changes in structure require changes in all application programs
- Implementation limitations
- No data definition
- Lack of standards

Network Model

Advantages

- Conceptual simplicity
- Handles more relationship types
- Data access is flexible
- Data owner/member relationship promotes data integrity
- Conformance to standards
- Includes data definition language (DDL) and data manipulation language (DML)

Disadvantages

- System complexity limits efficiency
- Navigational system yields complex implementation, application development, and management
- Structural changes require changes in all application programs

Relational Model

Advantages

- Structural independence is promoted using independent tables
- Tabular view improves conceptual simplicity
- Ad hoc query capability is based on SQL
- Isolates the end user from physical-level details
- Improves implementation and management simplicity

Disadvantages

- Requires substantial hardware and system software overhead
- Conceptual simplicity gives untrained people the tools to use a good system poorly
- May promote information problems

Entity Relationship Model

Advantages

- Visual modeling yields conceptual simplicity
- Visual representation makes it an effective communication tool
- Is integrated with the dominant relational model

Disadvantages

- Limited constraint representation
- Limited relationship representation
- No data manipulation language
- Loss of information content occurs when attributes are removed from entities to avoid crowded displays

Object-Oriented Model

Advantages

- Semantic content is added
- Visual representation includes semantic content
- Inheritance promotes data integrity

Disadvantages

- Slow development of standards caused vendors to supply their own enhancements
    - Compromised widely accepted standard
- Complex navigational system
- Learning curve is steep
- High system overhead slows transactions

NoSQL

Advantages	Disadvantages
• High scalability, availability, and fault tolerance are provided • Uses low-cost commodity hardware • Supports Big Data • Key-value model improves storage efficiency	• Complex programming is required • There is no relationship support • There is no transaction integrity support • In terms of data consistency, it provides an eventually consistent model