Week # 6

Database Management System

Examples of Business Rules

- A customer may generate many invoices.
- An invoice is generated by only one customer.
- A training session cannot be scheduled for fewer than 10 employees or for more than 30 employees.

Examples of Business Rules

- Note that those business rules establish entities, relationships, and constraints.
- **For example:** the first two business rules establish two entities (CUSTOMER and INVOICE) and a 1:M relationship between those two entities.
- The third business rule establishes a **constraint** (no fewer than 10 people and no more than 30 people), two **entities** (EMPLOYEE and TRAINING), and a **relationship** between EMPLOYEE and TRAINING.

Discovering Business Rules

- The main sources of business rules are company managers, policy makers, department managers, and written documentations such as a company’s procedures, standards or operations manuals**.**
- **A faster and more direct source of business rules is direct interviews with end users.**

Discovering Business Rules

- Unfortunately, because perceptions differ, end users sometimes are a <u>less reliable source</u> when it comes to specifying business rules.
- **For example,** a maintenance department mechanic might believe that any mechanic can initiate a maintenance procedure, when actually only mechanics with inspection authorization can perform such a task.

The process of identifying and documenting business rules is essential to database design for several reasons:

- They help <u>standardize</u> the company's view of data.
- They can be a <u>communications tool</u> between users and designers.
- They allow the designer to <u>understand the nature, role, and scope of the data</u>.
- They allow the designer to <u>understand business processes</u>.
- They allow the designer to develop appropriate relationship participation rules, constraints and to <u>select an accurate data model</u>.

Another Example:

- Of course, not all business rules can be modeled.
- For example, a business rule that specifies that “no pilot can fly more than 10 hours within any 24-hour period” cannot be modeled.
- However, such a business rule can be enforced by application software.

Translating Business Rules into Data Model Components

- Business rules set the stage for the proper identification of entities, attributes, relationships, and constraints.
- In the real world, names are used to identify objects. If the business environment wants to keep track of the objects, there will be specific business rules for them.
- As a general rule, a noun in a business rule will translate into an entity in the model, and a verb (active or passive) associating nouns will translate into a relationship among the entities.

Translating Business Rules into Data Model Components

- For example, the business rule “a customer may generate many invoices” contains two nouns (*customer and invoices) and a verb* (*generate) that associates the nouns. From this business rule, you could deduct that:*
- Customer and invoice are objects of interest for the environment and should be represented by their respective entities.
- There is a “generate” relationship between customer and invoice.

Translating Business Rules into Data Model Components

- To properly identify the type of relationship, you should consider that relationships are bidirectional; that is, they go both ways.
- **For example**, the business rule “a customer may generate many invoices” is complemented by the business rule “an invoice is generated by only one customer.”
- In that case, the relationship is one-to-many (1:M). Customer is the “1” side, and invoice is the “many” side.

Translating Business Rules into Data Model Components

- As a general rule, to properly identify the relationship type, you should ask two questions:
- How many instances of B are related to one instance of A?
- How many instances of A are related to one instance of B?

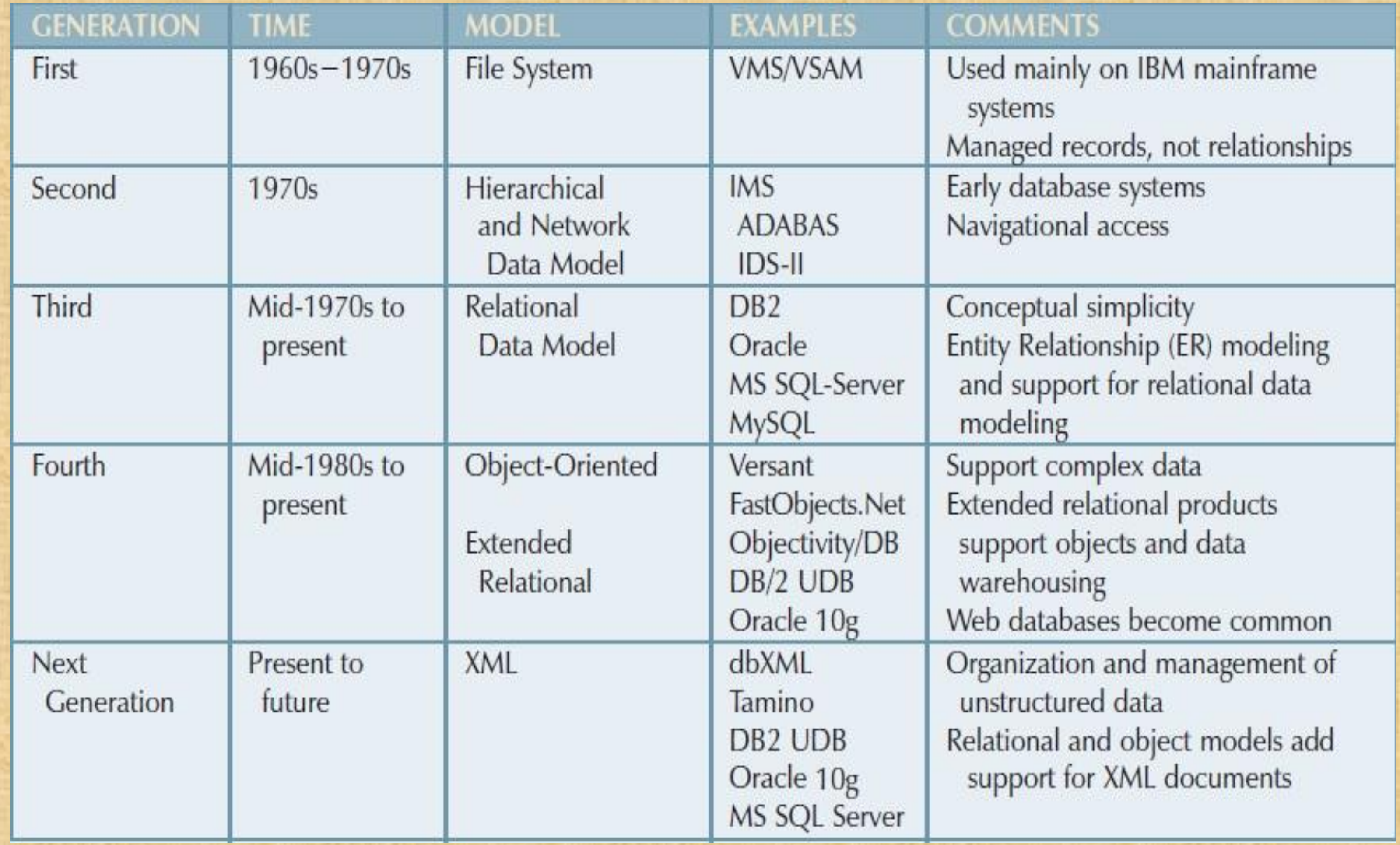
Translating Business Rules into Data Model Components

- For example, you can assess the relationship between student and class by asking two questions:
- In how many classes can one student enroll? Answer: many classes.
- How many students can enroll in one class? Answer: many students.

Translating Business Rules into Data Model Components

- Therefore, the relationship between student and class is many-to-many (M:N). You will have many opportunities to determine the relationships between entities as you proceed through given contents and soon the process will become second nature.

Evolution of Major Data Models



GENERATION	TIME	MODEL	EXAMPLES	COMMENTS
First	1960s–1970s	File System	VMS/VSAM	Used mainly on IBM mainframe systems Managed records, not relationships
Second	1970s	Hierarchical and Network Data Model	IMS ADABAS IDS-II	Early database systems Navigational access
Third	Mid-1970s to present	Relational Data Model	DB2 Oracle MS SQL-Server MySQL	Conceptual simplicity Entity Relationship (ER) modeling and support for relational data modeling
Fourth	Mid-1980s to present	Object-Oriented Extended Relational	Versant FastObjects.Net Objectivity/DB DB/2 UDB Oracle 10g	Support complex data Extended relational products support objects and data warehousing Web databases become common
Next Generation	Present to future	XML	dbXML Tamino DB2 UDB Oracle 10g MS SQL Server	Organization and management of unstructured data Relational and object models add support for XML documents

End of the Lecture

Network Model

The network model was created to represent **complex data relationships** more effectively than the hierarchical model, to improve database performance, and **to impose a database standard**.

Network Model

- The lack of **database standards** was troublesome to **programmers and application designers** because it made **database designs and applications less portable**.
- To help establish database standards, the Conference on **Data Systems Languages (CODASYL)** created the **Database Task Group (DBTG)** in the late **1960s**. The DBTG was charged to define standard specifications for an environment that would facilitate database creation and data manipulation.

Network Model

- The final DBTG report contained specifications for **three crucial database components:**
- The **schema**, which is the **conceptual organization of the entire database as viewed by the database administrator**. The schema includes a definition of the database name, the record type for each record, and the components that make up those records.

Network Model

- The **subschema**, which defines the portion of the database "***seen***" by the **application programs** that actually produce the desired information from the data contained within the database.
- The existence of subschema definitions allows all application programs to simply **invoke the subschema required to access the appropriate database file(s)**.
- A **data management language (DML)** that defines the environment in which data can be managed.

Network Model

- To produce the desired standardization for each of the three components, the **DBTG** specified three distinct DML components:
- A **schema data definition language (DDL),** which enables the database administrator to define the schema components.
- -A **subschema DDL,** which allows the application programs to define the database components that will be used by the application.
- -A **data manipulation language** to work with the data in the database.

Relational Model

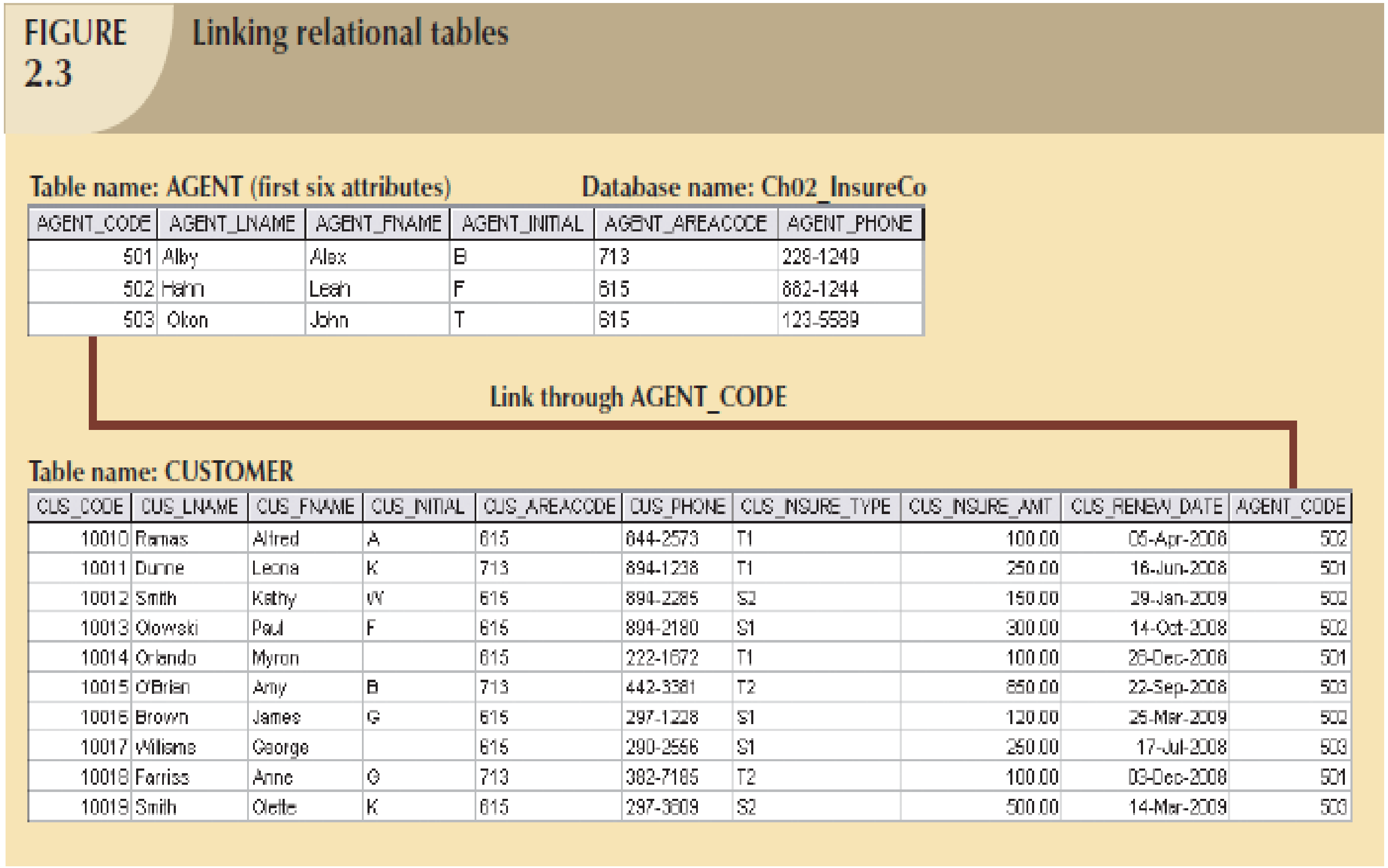


FIGURE 2.3 Linking relational tables

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

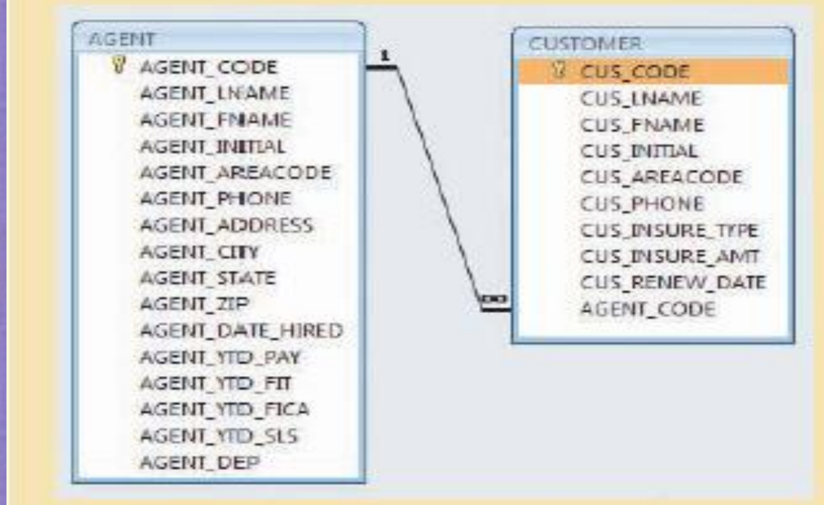
Link through AGENT_CODE

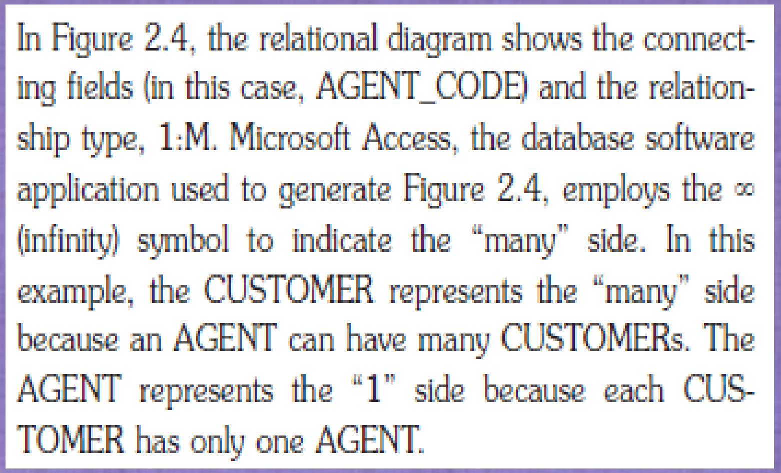
Table name: CUSTOMER

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-2008	502
10011	Dunne	Leona	K	713	894-1238	T1	250.00	16-Jun-2008	501
10012	Smith	Kathy	W	615	894-2285	S2	150.00	29-Jan-2009	502
10013	Olowski	Paul	F	615	894-2180	S1	300.00	14-Oct-2008	502
10014	Orlando	Myron		615	222-1672	T1	100.00	28-Dec-2008	501
10015	O'Brian	Amy	B	713	442-3381	T2	850.00	22-Sep-2008	503
10016	Brown	James	G	615	297-1228	S1	120.00	25-Mar-2009	502
10017	Williams	George		615	290-2556	S1	250.00	17-Jul-2008	503
10018	Farriss	Anne	G	713	382-7185	T2	100.00	03-Dec-2008	501
10019	Smith	Olette	K	615	297-3809	S2	500.00	14-Mar-2009	503

Relational Diagram

FIGURE 2.4 A relational diagram




In Figure 2.4, the relational diagram shows the connecting fields (in this case, AGENT_CODE) and the relationship type, 1:M. Microsoft Access, the database software application used to generate Figure 2.4, employs the ∞ (infinity) symbol to indicate the "many" side. In this example, the CUSTOMER represents the "many" side because an AGENT can have many CUSTOMERs. The AGENT represents the "1" side because each CUSTOMER has only one AGENT.

The Entity Relationship Model

- The **conceptual simplicity** of relational database technology triggered the demand for RDBMSs.
- Need for **more complex database implementation structures**.
- Creating the need for **more effective database design tools**.
- Database designers prefer to use a **graphical tool** in which **entities and their relationships** are pictured.

The Entity Relationship Model

- The entity relationship (ER) model, or ERM, has become a widely accepted standard for data modeling.
- Complemented the relational data model concepts.
- ER models are normally represented in an entity relationship diagram (ERD), which uses graphical representations to model database components.

The Entity Relationship Model

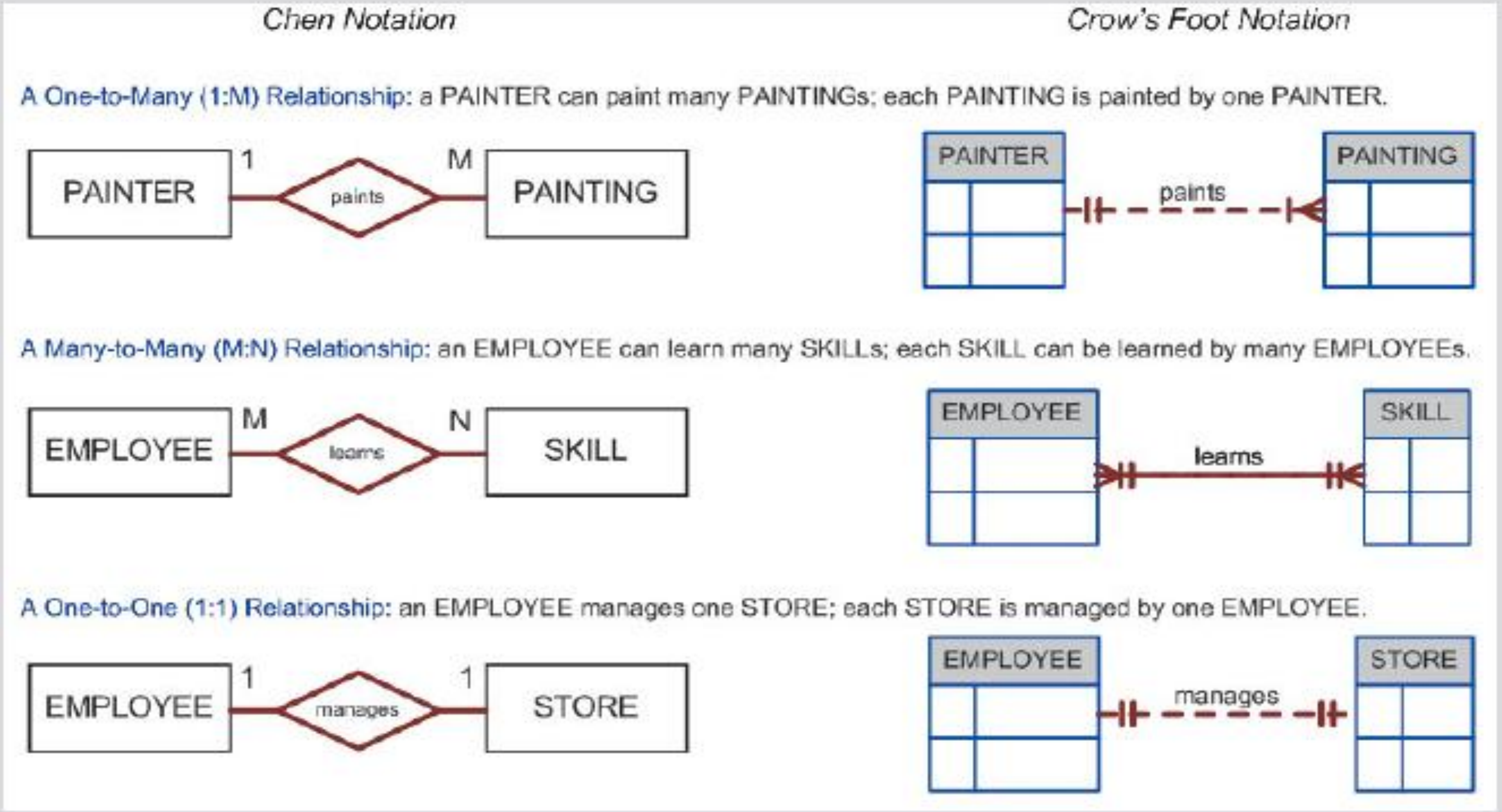
- The ER model is based on the following components:
- **Entity**. An entity was defined as **anything** about which data are to be collected and stored. An entity is represented in the ERD by a **rectangle**, also known as an entity box. The name of the entity, a **noun**, is written in the center of the rectangle. The entity name is generally written in **capital letters** and is written in the **singular form**: **PAINTER rather than PAINTERS**, and EMPLOYEE rather than EMPLOYEES.
- Usually, when applying the ERD to the relational model, an entity is mapped to a relational table. Each row in the relational table is known as an **entity instance** or **entity occurrence** in the ER model**.**

The Entity Relationship Model

- **Relationships**: Relationships describe **associations** among data. Most relationships describe associations between two entities. When the basic data model components were introduced, three types of relationships among data were illustrated: **one-to-many (1:M), many-to-many (M:N), and one-to-one (1:1)**.
- The ER model uses the term **connectivity** to label the relationship types. The name of the relationship usually is an **active or passive verb**. For example, a **PAINTER paints many PAINTINGs**; an EMPLOYEE learns many SKILLs; an EMPLOYEE manages a STORE.

Different types of relationships using two ER notations: the original Chen notation and the more current Crow's Foot notation:

FIGURE 2.5 The Chen and Crow's Foot notations



A One-to-Many (1:M) Relationship: a PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER.

A Many-to-Many (M:N) Relationship: an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs.

A One-to-One (1:1) Relationship: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE.

The Object-Oriented (OO) Model

- Increasingly **complex real-world problems** demonstrated a need for a data model that more closely represented the real world.
- In the object-oriented data model (OODM), **both data and their relationships are contained in a single structure known as an object**. In turn, the OODM is the basis for the **object-oriented database management system (OODBMS).**

The Object-Oriented (OO) Model

- An OODM reflects a very different way to define and use entities. **An object includes information about relationships between the facts within the object, as well as information about its relationships with other objects.**
- Therefore, the facts within the object are given greater meaning. The OODM is said to be a **semantic data model** because semantic indicates meaning.

The Object-Oriented (OO) Model

- **Attributes** describe the properties of an object. For example, a PERSON object includes the attributes Name, Social Security Number, and Date of Birth.
- Objects that share similar characteristics are grouped in classes. A **class** is a collection of similar objects with shared **structure (attributes)** and **behavior (methods)**. In a general sense, a class resembles the ER model’s entity set. However, a class is different from an entity set in that it contains **a set of procedures known as methods**.

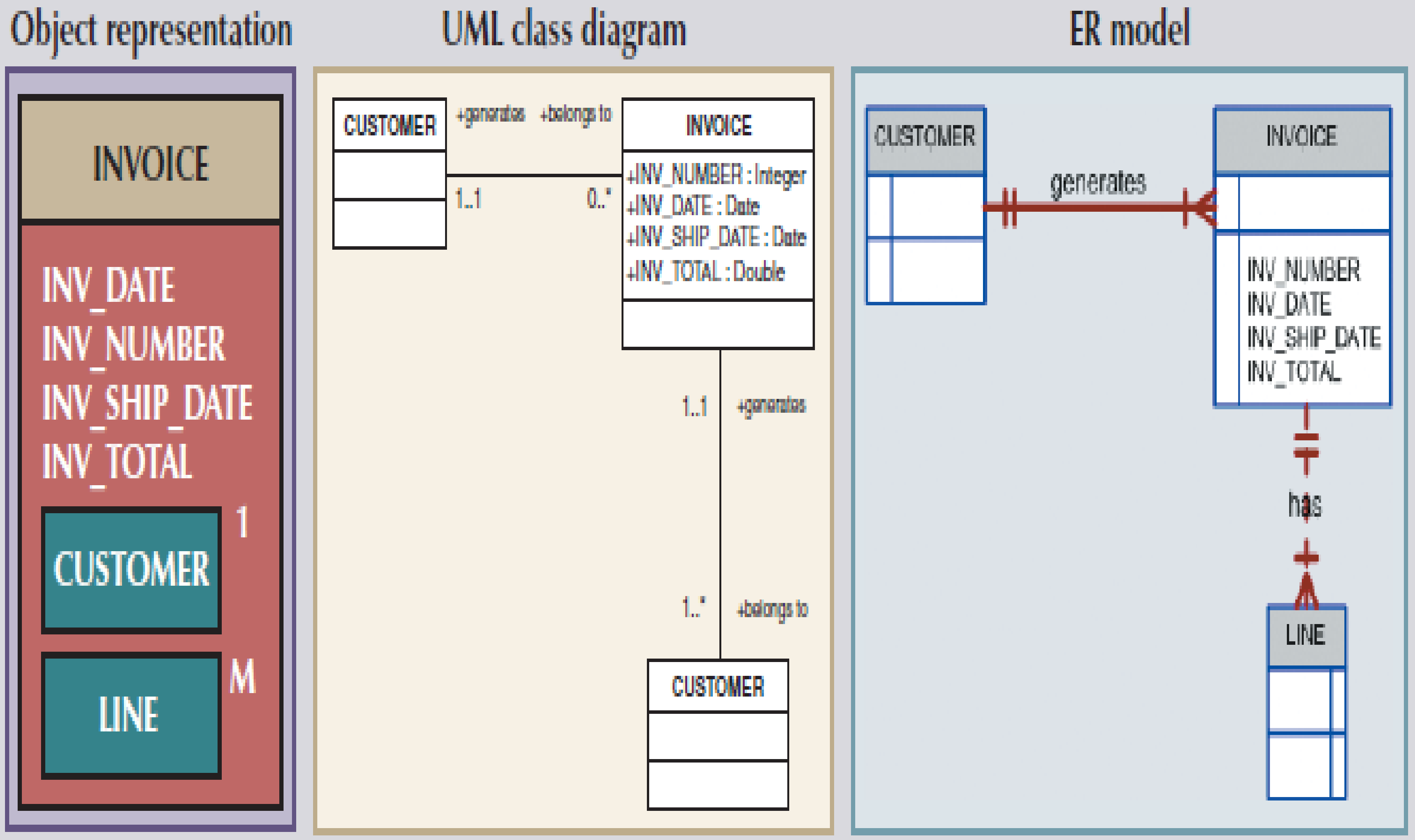
The Object-Oriented (OO) Model

- **Inheritance** is the ability of an object within the class hierarchy to **inherit the attributes and methods of the classes above it**.
- For example, two classes, CUSTOMER and EMPLOYEE, can be created as subclasses from the class PERSON. In this case, CUSTOMER and EMPLOYEE will inherit all attributes and methods from PERSON.

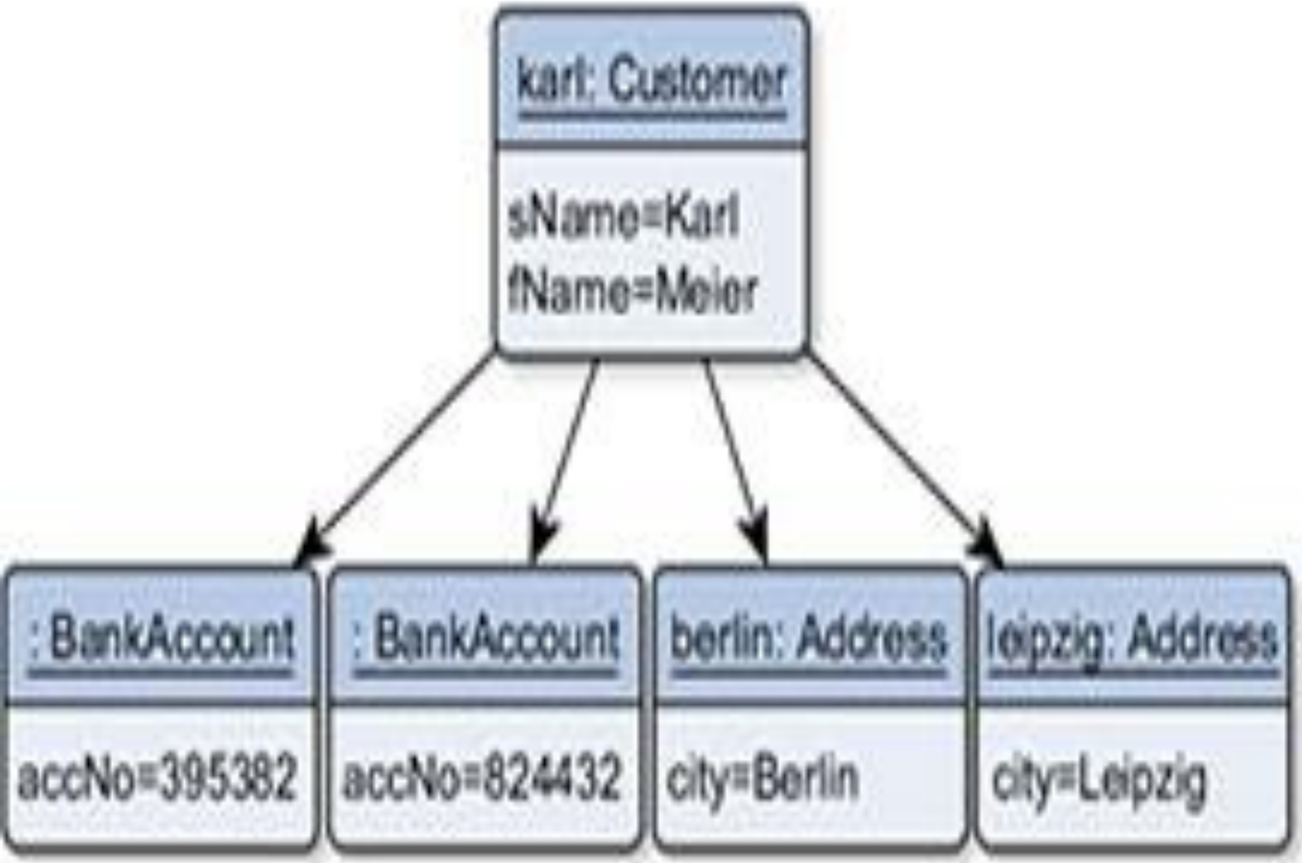
The Object-Oriented (OO) Model

- Object-oriented data models are typically depicted using **Unified Modeling Language (UML) class diagrams**.
- Unified Modeling Language (UML) is a language **based on OO concepts that describes a set of diagrams and symbols that can be used to graphically model a system.** UML class diagrams are used to represent data and their relationships within the larger UML object-oriented systems modeling language.

FIGURE 2.6 A comparison of OO, UML, and ER models



Object-Oriented Data Model



Relational Data Model

Customers

ID	SName	FName
1	Karl	Meier

Addresses

ID	City	CID
1	Berlin	1
2	Leipzig	1

BankAccounts

ID	AccNo	CID
1	395382	1
2	824432	1

The Convergence of Data Models

- Another semantic data model was developed in response to the increasing complexity of applications—the **extended relational data model (ERDM).** The ERDM, championed by many relational database researchers, constitutes the relational model's response to the OODM. This model includes **many of the OO model's best features within an inherently simpler relational database structural environment**. That's why a DBMS based on the ERDM is often described as an **object/relational database management system (O/RDBMS).**

The Convergence of Data Models

- With the huge installed base of the relational database and the emergence of the ERDM, the OODM faces an uphill battle. Although the ERDM includes a strong **semantic component**, it is primarily based on the relational data model's concepts.
- In contrast, the **OODM is wholly based on the OO and semantic data model concepts**. The **ERDM** is primarily geared to **business applications**, while the **OODM** tends to focus on **very specialized engineering and scientific applications**.

The Convergence of Data Models

- In the database arena, the most likely scenario appears to be an ever-increasing merging of OO and relational data model concepts and procedures, with an increasing emphasis on data models that **facilitate Internet-age technologies**.

Database Models and the Internet

- The use of the **Internet as a prime business tool** has drastically changed the role and scope of the database market.
- In fact, the Internet's impact on the database market has generated **new database product strategies** in which the OODM and ERDM-O/RDM have taken a backseat to **Internet-age database development**.

Database Models and the Internet

- If the database fits well into the Internet picture, its precise modeling heritage is of relatively little consequence. That’s why the relational model has prospered by **incorporating components from other data models**.
- For example, Oracle Corporation’s **Oracle 10g** database contains **OO components within a relational database structure**, as does IBM’s current **DB2 version**.

Database Models and the Internet

- With the dominance of the World Wide Web, there is a **growing need to manage unstructured data**, such as the data found in most of today's documents and Web pages. In response to this need, current databases now support Internet-age technologies such as **Extensible Markup Language (XML)**.
- For example, extended relational databases such as Oracle 10g and IBM's DB2 support XML data types **to store and manage unstructured data**.

Data Models: A Summary

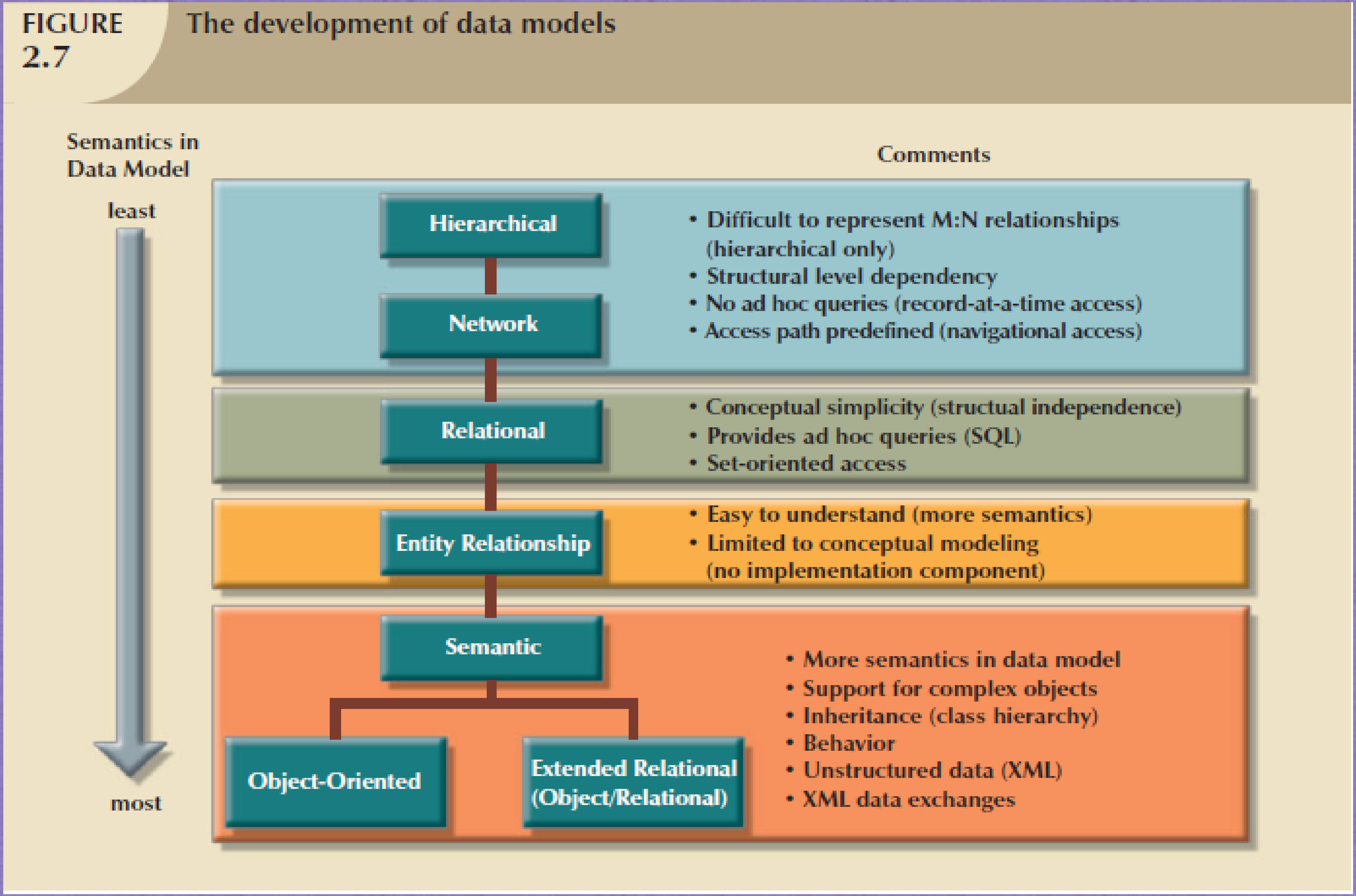


FIGURE 2.7 The development of data models

- Difficult to represent M:N relationships (hierarchical only)
- Structural level dependency
- No ad hoc queries (record-at-a-time access)
- Access path predefined (navigational access)

- Conceptual simplicity (structual independence)
- Provides ad hoc queries (SQL)
- Set-oriented access

- Easy to understand (more semantics)
- Limited to conceptual modeling (no implementation component)

- More semantics in data model
- Support for complex objects
- Inheritance (class hierarchy)
- Behavior
- Unstructured data (XML)
- XML data exchanges

TABLE 2.2 **Advantages and Disadvantages of Various Database Models**

DATA MODEL	DATA INDEPENDENCE	STRUCTURAL INDEPENDENCE	ADVANTAGES	DISADVANTAGES
Hierarchical	Yes	No	1. It promotes data sharing. 2. Parent/Child relationship promotes conceptual simplicity. 3. Database security is provided and enforced by DBMS. 4. Parent/Child relationship promotes data integrity. 5. It is efficient with 1:M relationships.	1. Complex implementation requires knowledge of physical data storage characteristics. 2. Navigational system yields complex application development, management, and use; requires knowledge of hierarchical path. 3. Changes in structure require changes in all application programs. 4. There are implementation limitations (no multiparent or M:N relationships). 5. There is no data definition or data manipulation language in the DBMS. 6. There is a lack of standards.
Network	Yes	No	1. Conceptual simplicity is at least equal to that of the hierarchical model. 2. It handles more relationship types, such as M:N and multiparent. 3. Data access is more flexible than in hierarchical and file system models. 4. Data Owner/Member relationship promotes data integrity. 5. There is conformance to standards. 6. It includes data definition language (DDL) and data manipulation language (DML) in DBMS.	1. System complexity limits efficiency—still a navigational system. 2. Navigational system yields complex implementation, application development, and management. 3. Structural changes require changes in all application programs.
Relational	Yes	Yes	1. Structural independence is promoted by the use of independent tables. Changes in a table's structure do not affect data access or application programs. 2. Tabular view substantially improves conceptual simplicity, thereby promoting easier database design, implementation, management, and use. 3. Ad hoc query capability is based on SQL. 4. Powerful RDBMS isolates the end user from physical-level details and improves implementation and management simplicity.	1. The RDBMS requires substantial hardware and system software overhead. 2. Conceptual simplicity gives relatively untrained people the tools to use a good system poorly, and if unchecked, it may produce the same data anomalies found in file systems. 3. It may promote "islands of information" problems as individuals and departments can easily develop their own applications.
Entity Relationship	Yes	Yes	1. Visual modeling yields exceptional conceptual simplicity. 2. Visual representation makes it an effective communication tool. 3. It is integrated with dominant relational model.	1. There is limited constraint representation. 2. There is limited relationship representation. 3. There is no data manipulation language. 4. Loss of information content occurs when attributes are removed from entities to avoid crowded displays. (This limitation has been addressed in subsequent graphical versions.)
Object-Oriented	Yes	Yes	1. Semantic content is added. 2. Visual representation includes semantic content. 3. Inheritance promotes data integrity.	1. Slow development of standards caused vendors to supply their own enhancements, thus eliminating a widely accepted standard. 2. It is a complex navigational system. 3. There is a steep learning curve. 4. High system overhead slows transactions.

Note: All databases assume the use of a common data pool within the database. Therefore, *all* database models promote data sharing, thus eliminating the potential problem of islands of information.

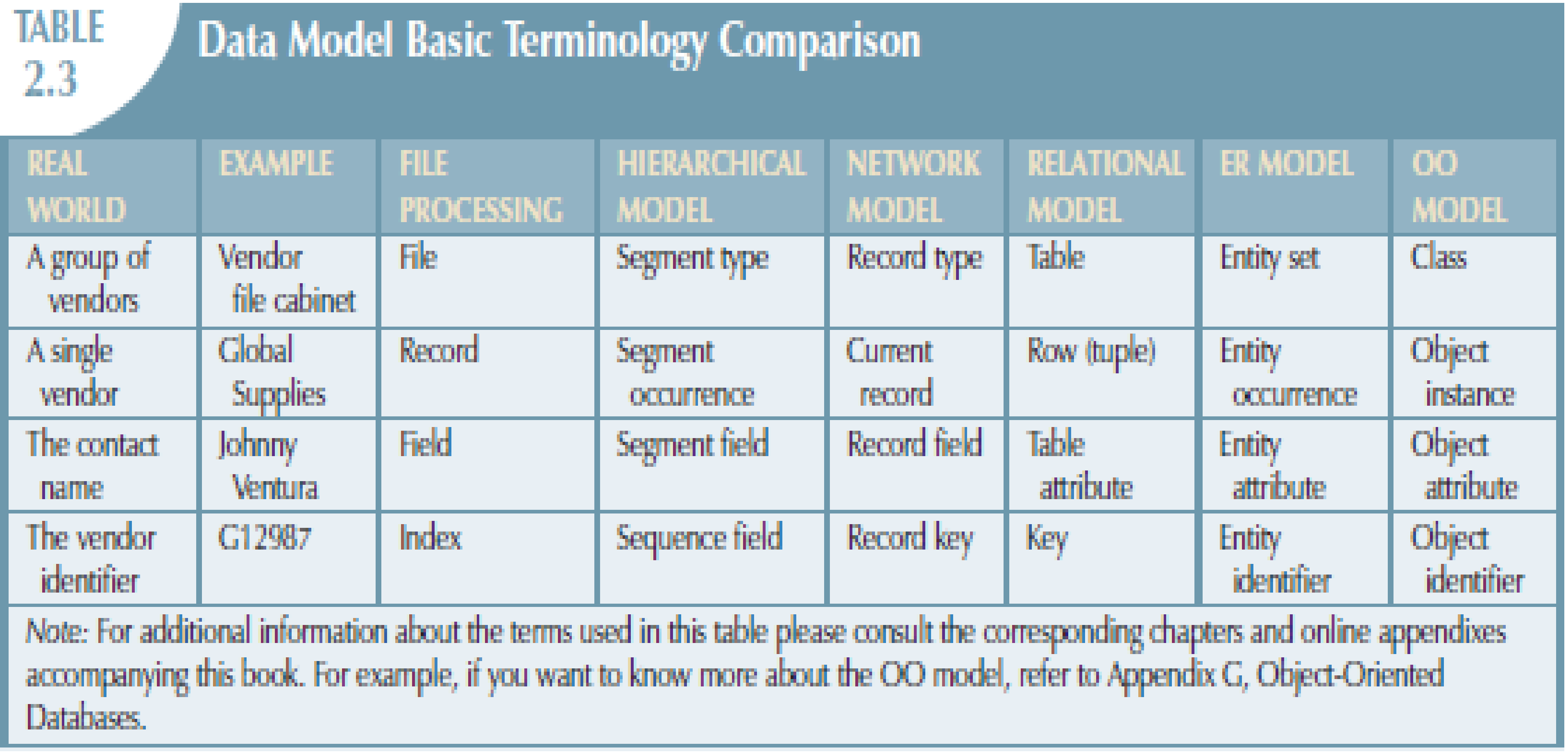


TABLE 2.3 Data Model Basic Terminology Comparison

REAL WORLD	EXAMPLE	FILE PROCESSING	HIERARCHICAL MODEL	NETWORK MODEL	RELATIONAL MODEL	ER MODEL	OO MODEL
A group of vendors	Vendor file cabinet	File	Segment type	Record type	Table	Entity set	Class
A single vendor	Global Supplies	Record	Segment occurrence	Current record	Row (tuple)	Entity occurrence	Object instance
The contact name	Johnny Ventura	Field	Segment field	Record field	Table attribute	Entity attribute	Object attribute
The vendor identifier	G12987	Index	Sequence field	Record key	Key	Entity identifier	Object identifier

Note: For additional information about the terms used in this table please consult the corresponding chapters and online appendixes accompanying this book. For example, if you want to know more about the OO model, refer to Appendix G, Object-Oriented Databases.

End of the Lecture