Database Management Systems

Week No. 5

Data Independence Levels

- Data independence is normally thought of in terms of two levels or types.
- Logical data independence makes it possible to change the structure of the data independently of modifying the applications or programs that make use of the data. There is no need to rewrite current applications as part of the process of adding to or removing data from then system.

Data Independence Levels

- The second type or level of data independence is known as physical data independence.
- This approach has to do with altering the organization or storage procedures related to the data, rather than modifying the data itself.

Data Independence Levels

- Accomplishing this shift in file organization or the indexing strategy used for the data does not require any modification to the external structure of the applications, meaning that users of the applications are not likely to notice any difference at all in the function of their programs.

Data Independence Levels

- The relational database model gives us the luxury of forgetting the actual physical data storage characteristics, thereby allowing us to concentrate on the logical view of the database.
- That is, we may focus on the human perception of data storage rather than on the often difficult-to-comprehend manner in which the computer sees those same data.

Data Independence Levels

- Since the relational model achieves both data independence and structural independence, it becomes much easier to design the database and to manage its contents.

Database Models

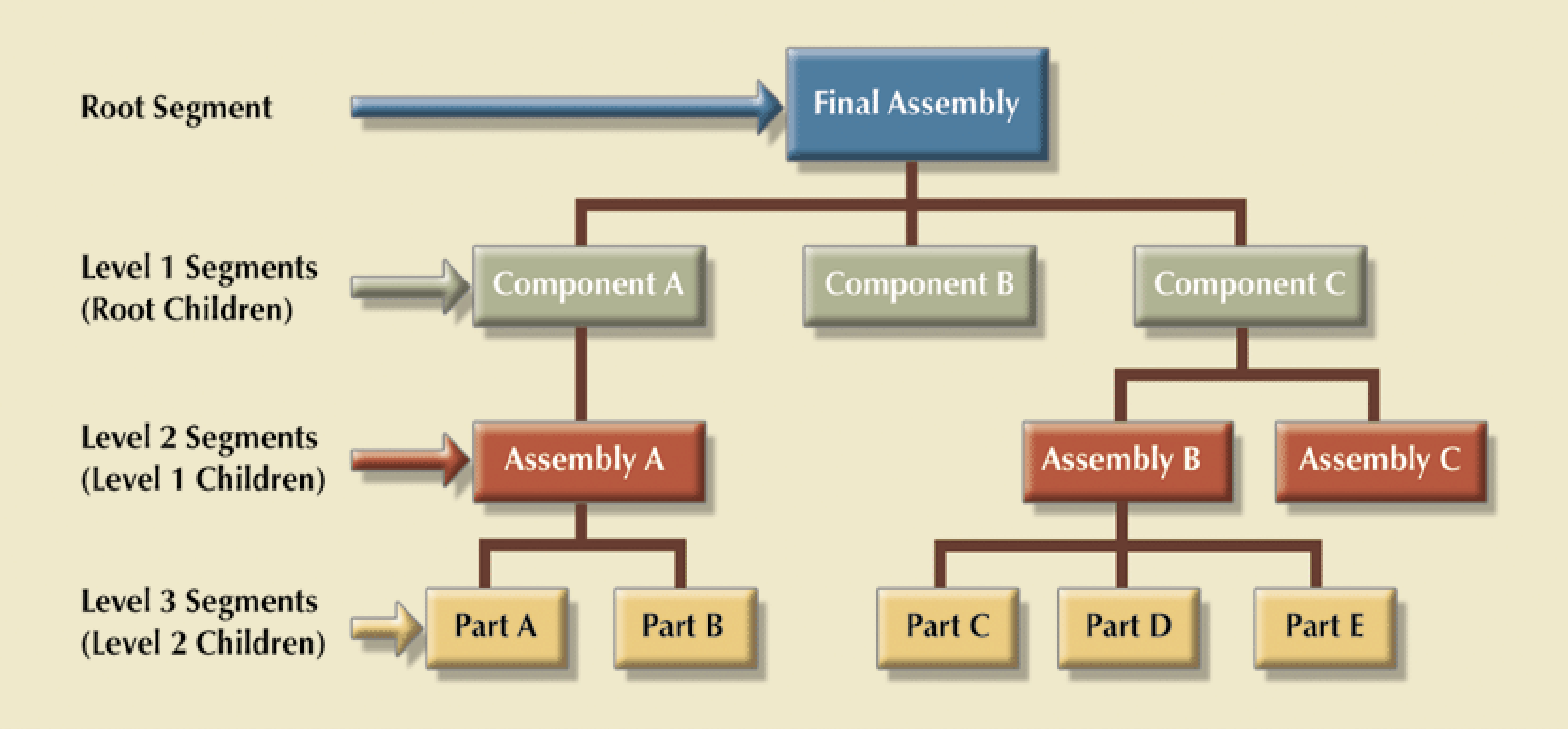
- Collection of logical constructs used to represent data structure and relationships within the database.
  - **Conceptual models:** logical nature of data representation.
  - **Implementation models:** emphasis on how the data are represented in the database.

Database Models: Historic Overview

- Flat files - 1960s - 1980s
- Hierarchical – 1970s - 1990s
- Network – 1970s - 1990s
- Relational – 1980s - present
- Object-oriented – 1990s - present
- Object-relational – 1990s - present
- Data warehousing – 1980s - present
- Web-enabled – 1990s - present

Hierarchical Database Model

- Logically represented by an upside down tree
  - Each parent can have many children
  - Each child has only one parent

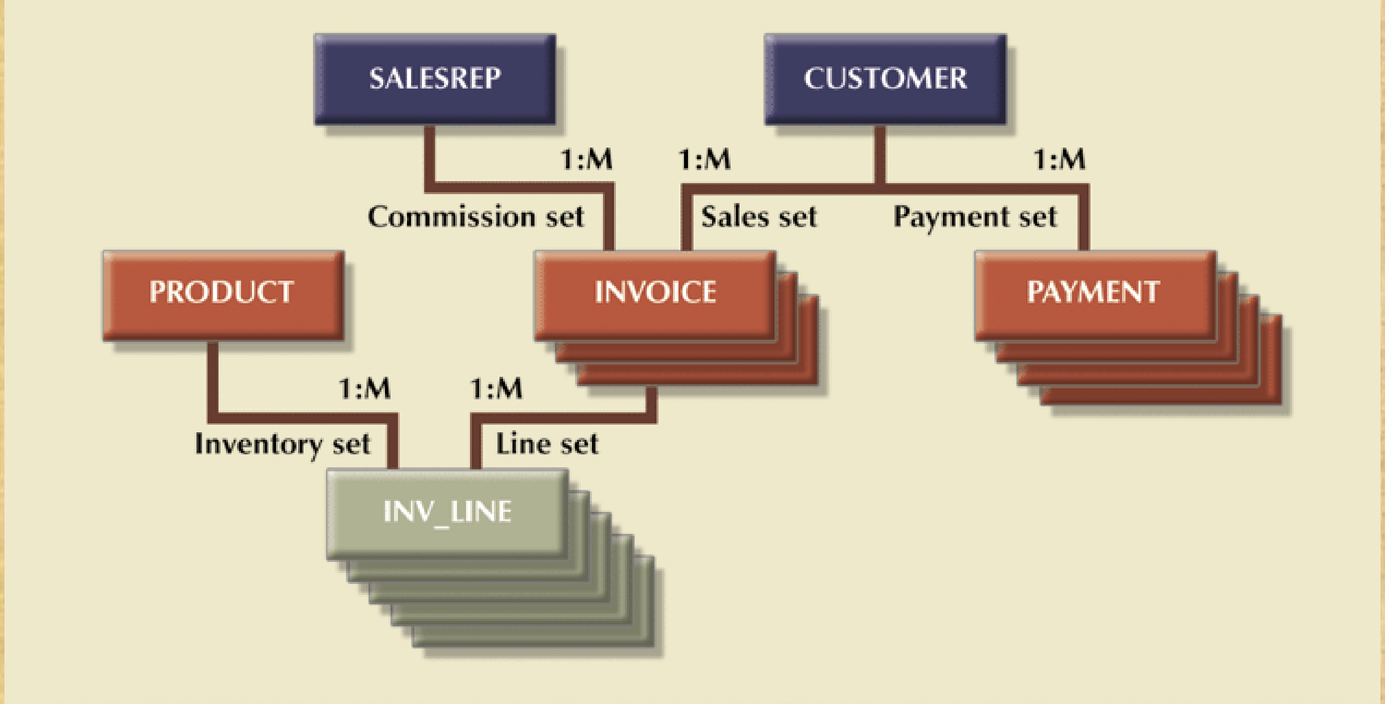


Hierarchical Database Model (Cont’d)

- **Advantages**
  - Conceptual simplicity
  - Database security and integrity
  - Data independence
  - Efficiency
- **Disadvantages**
  - Complex implementation
  - Difficult to manage and lack of standards
  - Lacks structural independence
  - Application programming and use complexity
  - Implementation limitations

Network Database Model

- Each record can have multiple parents
  - Composed of sets
  - Each set has owner record and member record
  - Member may have several owners



Network Database Model (Cont’d)

- **Advantages**
  - Conceptual simplicity
  - Handles more relationship types
  - Data access flexibility
  - Promotes database integrity
  - Data independence
  - Conformance to standards
- **Disadvantages**
  - System complexity
  - Lack of structural independence

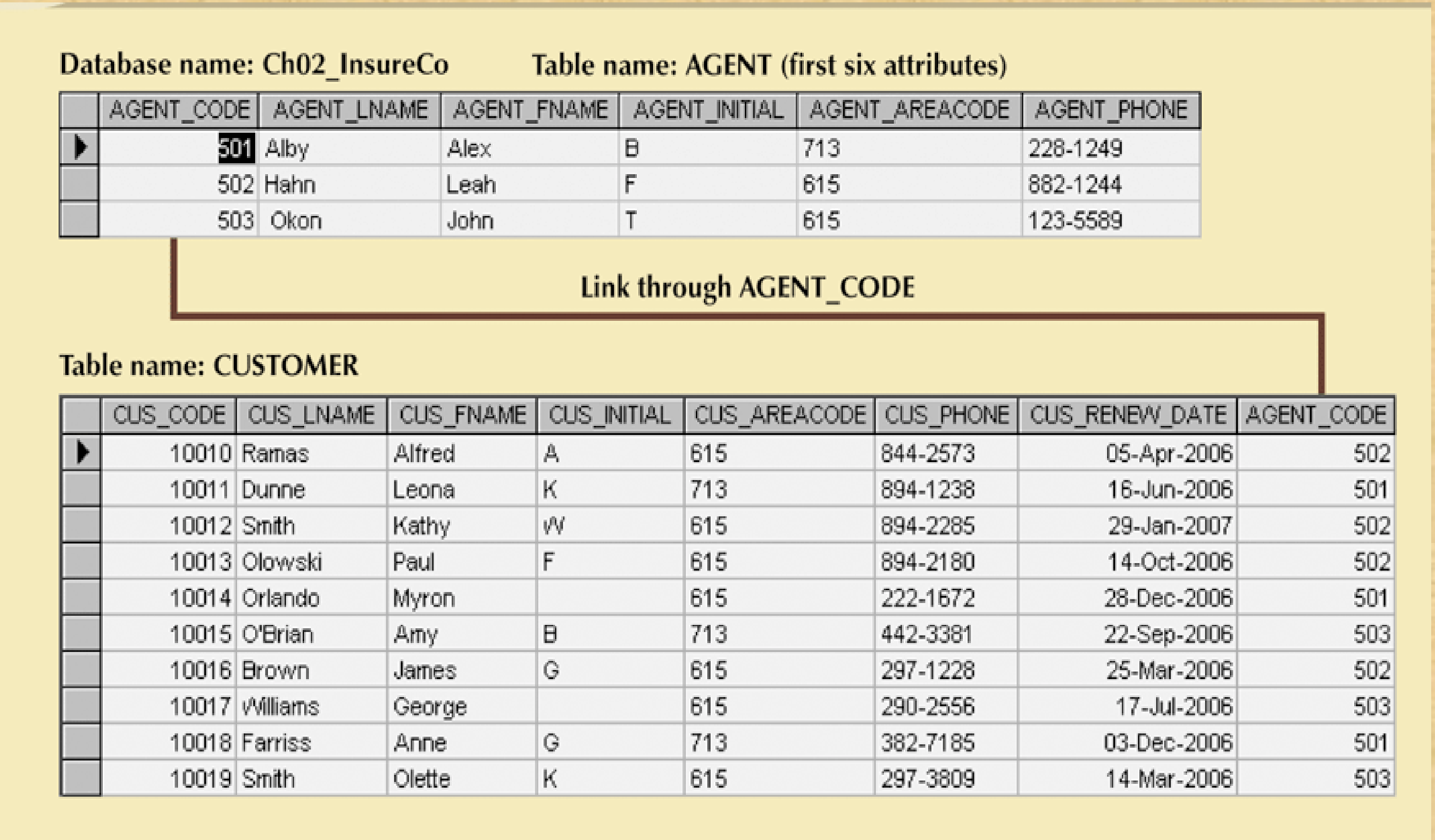
Relational Database Model

- Perceived by user as a collection of tables for data storage

- Tables are a series of row/column intersections

- Tables related by sharing common entity characteristic(s)

Relational Database Model (Cont’d)



Database name: Ch02_InsureCo

Table name: AGENT (first six attributes)

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

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_RENEW_DATE	AGENT_CODE
10010	Ramas	Alfred	A	615	844-2573	05-Apr-2006	502
10011	Dunne	Leona	K	713	894-1238	16-Jun-2006	501
10012	Smith	Kathy	W	615	894-2285	29-Jan-2007	502
10013	Olowski	Paul	F	615	894-2180	14-Oct-2006	502
10014	Orlando	Myron		615	222-1672	28-Dec-2006	501
10015	O'Brian	Amy	B	713	442-3381	22-Sep-2006	503
10016	Brown	James	G	615	297-1228	25-Mar-2006	502
10017	Williams	George		615	290-2556	17-Jul-2006	503
10018	Farriss	Anne	G	713	382-7185	03-Dec-2006	501
10019	Smith	Olette	K	615	297-3809	14-Mar-2006	503

Relational Database Model (Cont'd)

- Schema for the table
  - **Graphical representation**

AGENT

AGENT_CODE	AGENT_LNAME	AGENT_FNAME	AGENT_INITIAL	AGENT_AREACODE	AGENT_PHONE

  - **Text description**

AGENT(AGENT_CODE, AGENT_LNAME, AGENT_FNAME, AGENT_INITIAL, AGENT_AREACODE, AGETN_PHONE)

Relational Database Model (Cont’d)

- **Advantages**
  - Structural independence
  - Improved conceptual simplicity
  - Easier database design, implementation, management, and use
  - Ad hoc query capability with SQL
  - Powerful database management system
- **Disadvantages**
  - Substantial hardware and system software overhead
  - Poor design and implementation is made easy
  - May promote “islands of information” problems

Isolated Databases: **"islands of information "**

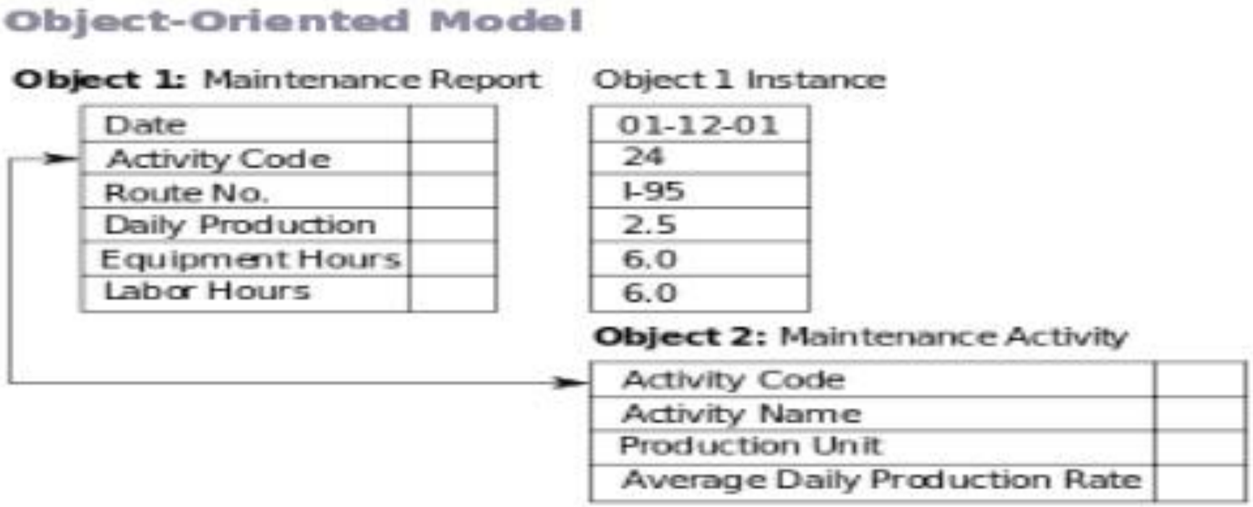
- Complex relational database systems can lead to these databases becoming "islands of information" where the **information cannot be shared easily from one large system to another**. Often, with big firms or institutions, you find relational databases grew in separate divisions differently.
- For example, maybe the hospital billing department used one database while the hospital personnel department used a different database. Getting those **databases to "talk" to each other** can be a large, and expensive, undertaking, yet in a complex hospital system, all the databases need to be involved for good patient and employee care.

Object-Oriented Database Model

- Objects or abstractions of real-world entities are stored
  - Attributes describe properties
  - Collection of similar objects is a class
    - Methods represent real world actions of classes
    - Classes are organized in a class hierarchy
  - Inheritance is ability of object to inherit attributes and methods of classes above it

OBJECT-ORIENTED DBMS

An object database (also object-oriented database management system) is a database management system in which information is represented in the form of objects as used in object-orientedprogramming. Object databases are different from relational databaseswhich are table-oriented.



Object-Oriented Model

Object 1: Maintenance Report

Date	
Activity Code	
Route No.	
Daily Production	
Equipment Hours	
Labor Hours	

Object 1 Instance

01-12-01
24
I-95
2.5
6.0
6.0

Object 2: Maintenance Activity

Activity Code	
Activity Name	
Production Unit	
Average Daily Production Rate	

OO Data Model

- **Advantages**
  - Adds semantic content
  - Visual presentation includes semantic content
  - Database integrity
  - Both structural and data independence
- **Disadvantages**
  - Lack of OODM
  - Complex navigational data access
  - Steep learning curve
  - High system overhead slows transactions

Costs and Risks of the Database Approach

- Up-front costs:
  - Installation Management Cost and Complexity
  - Conversion Costs
- Ongoing Costs
  - Requires New, Specialized Personnel
  - Need for Explicit Backup and Recovery
- Organizational Conflict

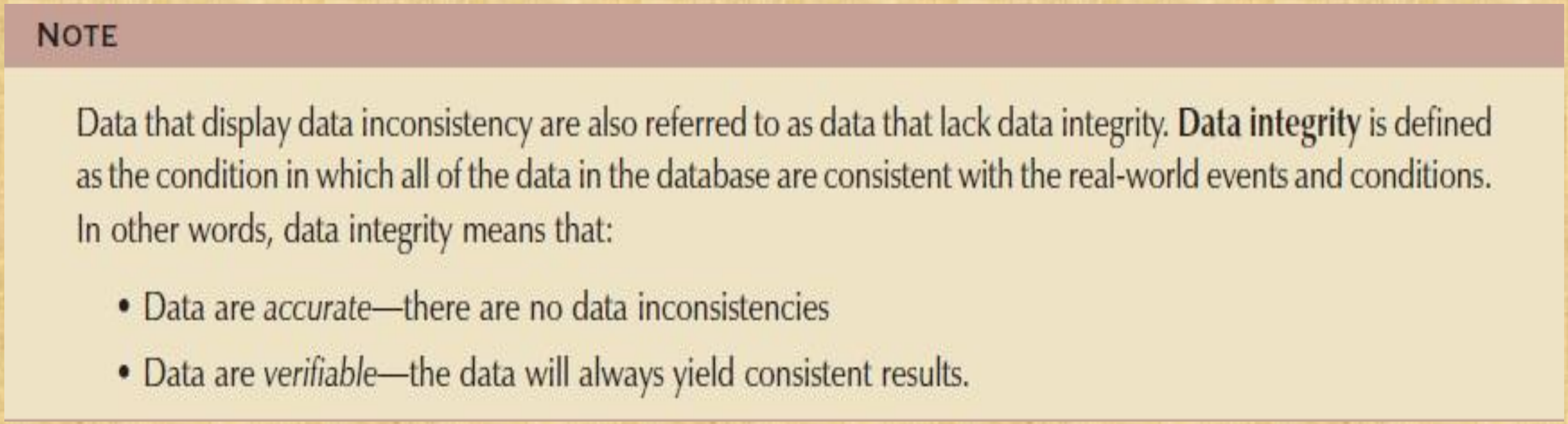
Review

- Basic concepts: data, information, database, DBMS, file, conceptual model, implementation model, etc
- Why database and its importance, cost and risk
- Different database models
  - definition
  - advantage
  - disadvantage

End of Lecture

Any Questions... ?

Data Integrity:



NOTE

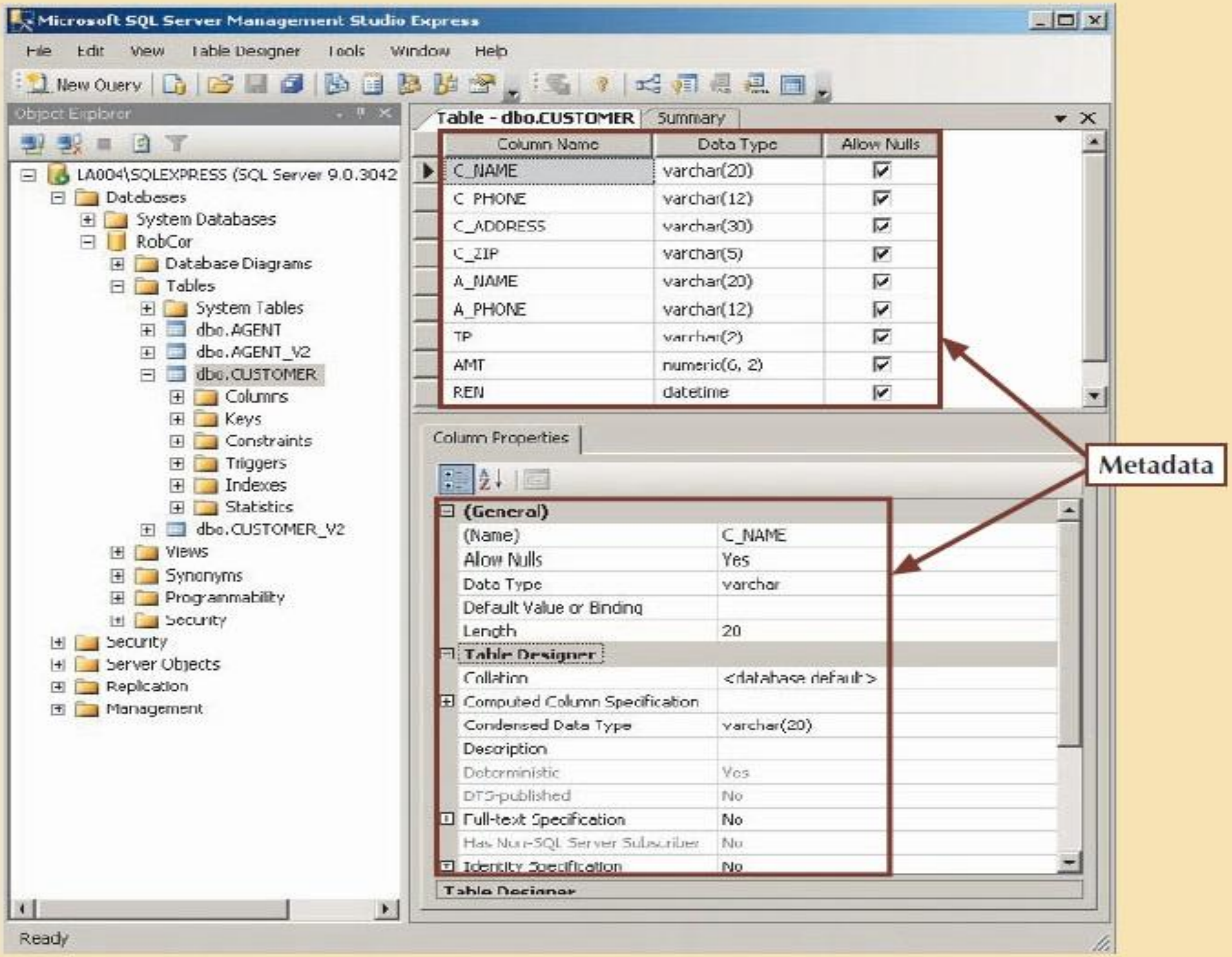
Data that display data inconsistency are also referred to as data that lack data integrity. **Data integrity** is defined as the condition in which all of the data in the database are consistent with the real-world events and conditions. In other words, data integrity means that:

- Data are *accurate*—there are no data inconsistencies
- Data are *verifiable*—the data will always yield consistent results.

Data storage management

- The DBMS creates and manages the complex structures required for data storage, thus relieving you from the difficult task of defining and programming the physical data characteristics.
- A modern DBMS provides storage not only for the data, but also for related data entry forms or screen definitions, report definitions, data validation rules, procedural code, structures to handle video and picture formats, and so on.

Illustrating metadata with Microsoft SQL Server Express



Column Name	Data Type	Allow Nulls
C_NAME	varchar(20)	☑
C_PHONE	varchar(12)	☑
C_ADDRESS	varchar(30)	☑
C_ZIP	varchar(5)	☑
A_NAME	varchar(20)	☑
A_PHONE	varchar(12)	☑
TP	varchar(2)	☑
AMT	numeric(6, 2)	☑
REN	datetime	☑

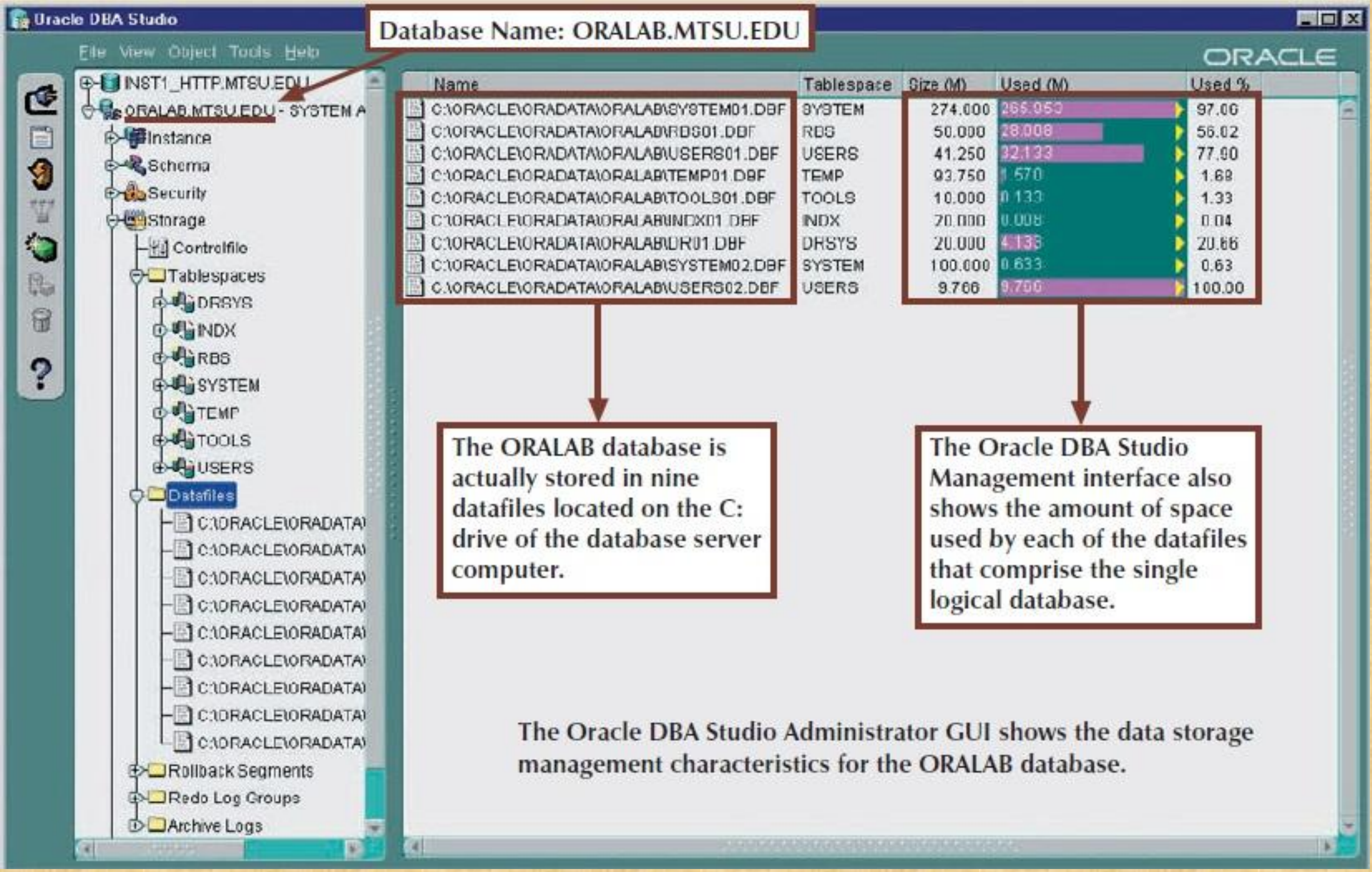
(General)	
(Name)	C_NAME
Allow Nulls	Yes
Data Type	varchar
Default Value or Binding	
Length	20
Table Designer	
Collation	<database default>
Computed Column Specification	
Condensed Data Type	varchar(20)
Description	
Deterministic	Yes
DTS-published	No
Full-text Specification	No
Has Non-SQL Server Subscriber	No
Identity Specification	No

Table Designer

Performance tuning

- Performance tuning relates to the activities that make the database perform more efficiently in terms of storage and access speed. Although the user sees the database as a single data storage unit, the DBMS actually stores the database in multiple physical data files.
- Such data files may even be stored on different storage media. Therefore, the DBMS doesn’t have to wait for one disk request to finish before the next one starts. In other words, the DBMS can fulfill database requests concurrently.

Illustrating data storage management with Oracle



Database Name: ORALAB.MTSU.EDU

Name	Tablespace	Size (M)	Used (M)	Used %
C:\ORACLE\ORADATA\ORALAB\SYSTEM01.DBF	SYSTEM	274.000	266.053	97.06
C:\ORACLE\ORADATA\ORALAB\RBS01.DBF	RBS	50.000	28.008	56.02
C:\ORACLE\ORADATA\ORALAB\USERS01.DBF	USERS	41.250	32.133	77.90
C:\ORACLE\ORADATA\ORALAB\TEMP01.DBF	TEMP	93.750	1.570	1.68
C:\ORACLE\ORADATA\ORALAB\TOOLS01.DBF	TOOLS	10.000	0.133	1.33
C:\ORACLE\ORADATA\ORALAB\INDX01.DBF	INDX	70.000	0.008	0.04
C:\ORACLE\ORADATA\ORALAB\DR01.DBF	DRSYS	20.000	4.133	20.66
C:\ORACLE\ORADATA\ORALAB\SYSTEM02.DBF	SYSTEM	100.000	0.633	0.63
C:\ORACLE\ORADATA\ORALAB\USERS02.DBF	USERS	9.766	9.700	100.00

The ORALAB database is actually stored in nine datafiles located on the C: drive of the database server computer.

The Oracle DBA Studio Management interface also shows the amount of space used by each of the datafiles that comprise the single logical database.

The Oracle DBA Studio Administrator GUI shows the data storage management characteristics for the ORALAB database.

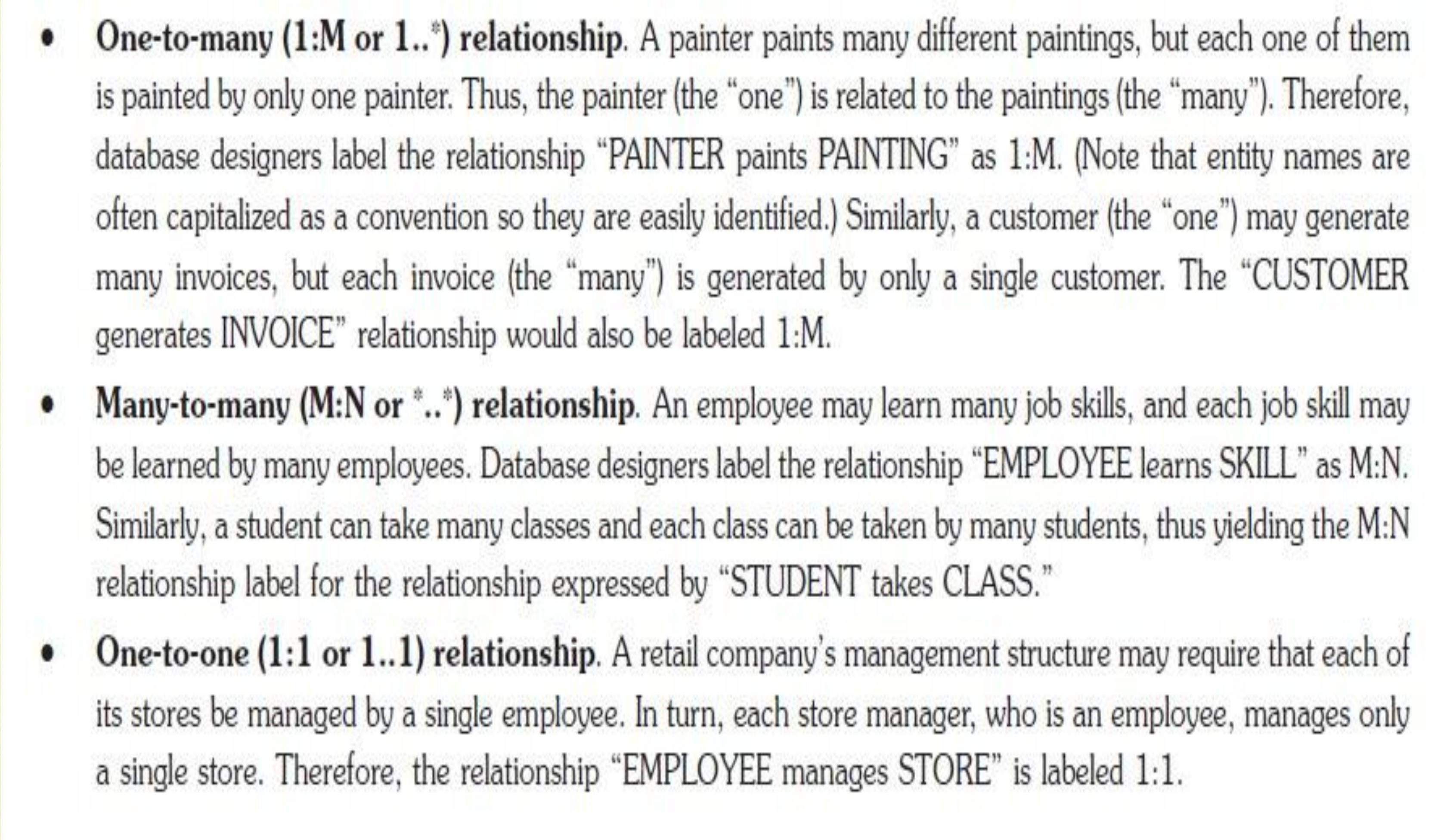
DATA MODEL BASIC BUILDING BLOCKS

- An **attribute is a characteristic of an entity. For example, a CUSTOMER entity would be described by attributes such** as customer last name, customer first name, customer phone, customer address, and customer credit limit. Attributes are the equivalent of fields in file systems.

DATA MODEL BASIC BUILDING BLOCKS (continued...)

- A **relationship describes an association among entities.**
- For example, a relationship exists between customers and agents that can be described as follows: an agent can serve many customers, and each customer may be served by one agent.

Data models use three types of relationships: one-to-many, many-to-many, and one-to-one.



- **One-to-many (1:M or 1..*) relationship**. A painter paints many different paintings, but each one of them is painted by only one painter. Thus, the painter (the "one") is related to the paintings (the "many"). Therefore, database designers label the relationship "PAINTER paints PAINTING" as 1:M. (Note that entity names are often capitalized as a convention so they are easily identified.) Similarly, a customer (the "one") may generate many invoices, but each invoice (the "many") is generated by only a single customer. The "CUSTOMER generates INVOICE" relationship would also be labeled 1:M.
- **Many-to-many (M:N or *..*) relationship**. An employee may learn many job skills, and each job skill may be learned by many employees. Database designers label the relationship "EMPLOYEE learns SKILL" as M:N. Similarly, a student can take many classes and each class can be taken by many students, thus yielding the M:N relationship label for the relationship expressed by "STUDENT takes CLASS."
- **One-to-one (1:1 or 1..1) relationship**. A retail company's management structure may require that each of its stores be managed by a single employee. In turn, each store manager, who is an employee, manages only a single store. Therefore, the relationship "EMPLOYEE manages STORE" is labeled 1:1.

Relationships are Bidirectional

- Each relationship in both directions; that is, relationships are bidirectional:

EXAMPLE:

- One CUSTOMER can generate many INVOICEs.
- Each of the many INVOICEs is generated by only one CUSTOMER

Constraint

- A **constraint is a restriction placed on the data. Constraints are important because they help to ensure data integrity.**
- Constraints are normally expressed in the form of rules.

For examples:

- An employee’s salary must have values that are between 6,000 and 350,000.
- A student’s GPA must be between 0.00 and 4.00.
- Each class must have one and only one teacher.

Business Rules

- **A business rule is a brief, precise, and unambiguous description of a policy, procedure, or principle within a specific organization.**
- In a sense, business rules are misnamed: they apply to any organization, large or small—a business, a government unit, a religious group, or a research laboratory-that stores and uses data to generate information.

Business Rules (continued...)

- Properly written business rules are used to define entities, attributes, relationships, and constraints.
- Business rules describe, in simple language, the main and distinguishing characteristics of the data *as viewed by the company*.

End of the Lecture