Chapter 4

Relational Databases

Learning Objectives

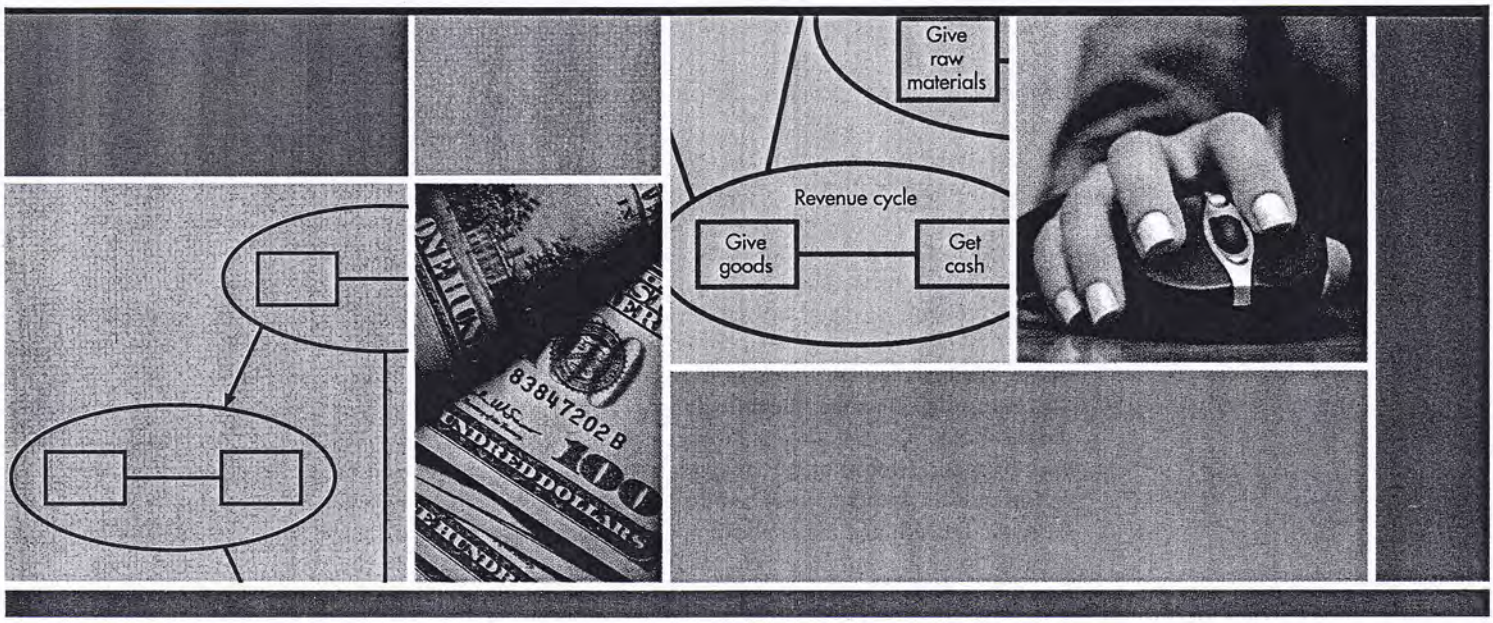
After studying this chapter, you should be able to:

1. Explain the importance and advantages of databases, as well as the difference between database systems and file-based legacy systems.
2. Explain the difference between logical and physical views of a database.
3. Explain fundamental concepts of database systems such as DBMS, schemas, the data dictionary, and DBMS languages.
4. Describe what a relational database is and how it organizes data.
5. Create a set of well-structured tables to store data in a relational database.
6. Perform simple queries using the Microsoft Access database.

INTEGRATIVE CASE S&S

S&S is very successful and operates five stores and a popular Web site. Ashton Fleming believes that it is time to upgrade S&S's AIS so that Susan and Scott can easily access the information they need to run their business. Most new AISs are based on a relational database. Since Ashton knows that Scott and Susan are likely to have questions, he prepared a brief report that explains why S&S's new AIS should be a relational database system. His report addresses the following questions:

1. What is a database system, and how does it differ from file-oriented systems?
2. What is a *relational* database system?
3. How do you design a well-structured set of tables in a relational database?
4. How do you query a relational database system?



Introduction

Relational databases underlie most modern integrated AISs. This chapter and Chapters 17 through 19 explain how to participate in the design and implementation of a database. This chapter defines a database, with the emphasis on understanding the relational database structure. Chapter 17 introduces two tools accountants use to design databases—entity-relationship diagramming and REA data modeling—and demonstrates how to use them to build an AIS data model. Chapter 18 explains how to implement an REA data model and how to produce the information needed to manage an organization. Chapter 19 discusses advanced data modeling and database design issues.

Files versus Databases

To appreciate the power of databases, it is important to understand how data are stored in computer systems. Figure 4-1 shows a data hierarchy. Information about the attributes of a customer, such as name and address, are stored in fields. All the fields containing data about one entity (e.g., one customer) form a record. A set of related records, such as all customer records, forms a file (e.g., the customer file). A set of interrelated, centrally coordinated files forms a ***database***.

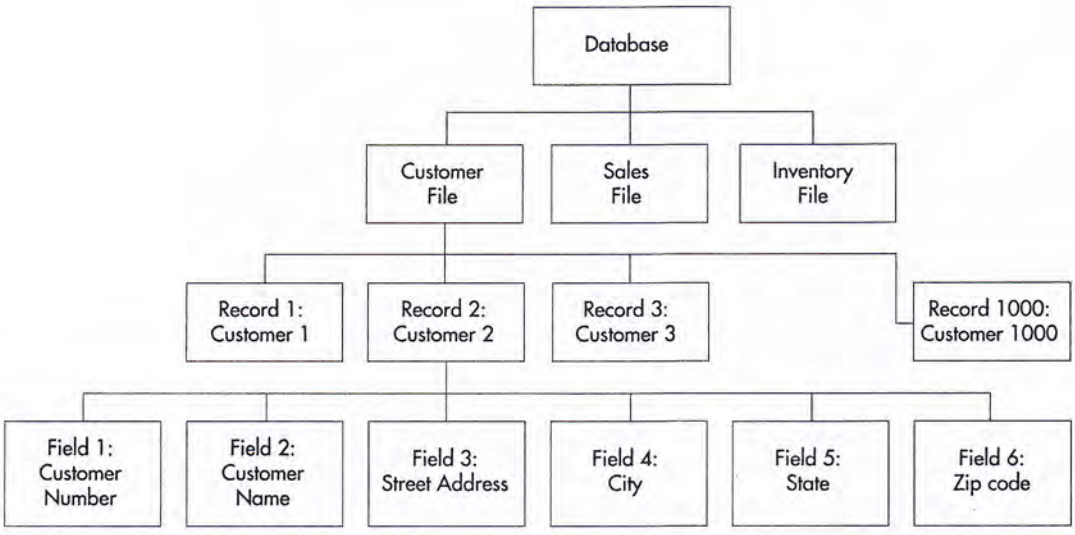


FIGURE 4-1
Basic Elements of Data Hierarchy

Databases were developed to address the proliferation of master files. For many years, companies created new files and programs each time a need for information arose. Bank of America once had 36 million customer accounts in 23 separate systems. This proliferation created problems such as storing the same data in two or more master files, as shown in Figure 4-2. This made it difficult to integrate and update data and to obtain an organization-wide view of data. It also created problems because the data in the different files were inconsistent. For example, a customer's address may have been correctly updated in the shipping master file but not the billing master file.

Figure 4-2 illustrates the differences between file-oriented systems and database systems. In the database approach, data is an organizational resource that is used by and managed for the entire organization, not just the originating department. A ***database management system (DBMS)*** is the interface between the database and the various application programs. The database, the DBMS, and the application programs that access the database through the DBMS are referred to as the ***database system***. The ***database administrator (DBA)*** is responsible for the database.

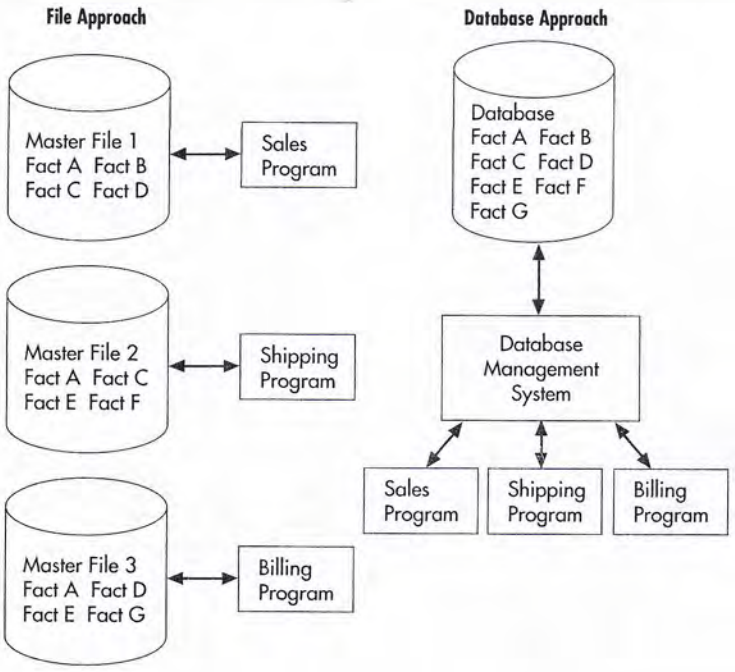
Using Data Warehouses for Business Intelligence

In today's fast-paced global economy, management must constantly reevaluate financial and operating performance in light of strategic goals and quickly alter plans as needed. Since strategic decision making requires access to large amounts of historical data, organizations are building separate databases called data warehouses. A ***data warehouse*** contains both detailed and summarized data for a number of years and is used for analysis rather than transaction processing. It is not unusual for data warehouses to contain hundreds of terabytes of data.

Data warehouses do not replace transaction processing databases; they complement them by providing support for strategic decision making. Since data warehouses are not used for transaction processing, they are usually updated periodically rather than in real time. Whereas transaction processing databases minimize redundancy and maximize the efficiency of updating them to reflect the results of current transactions, data warehouses are purposely redundant to maximize query efficiency.

Using a data warehouse for strategic decision making is often referred to as ***business intelligence***. There are two main techniques used in business intelligence: online analytical processing (OLAP) and data mining. ***Online analytical processing (OLAP)*** is using queries to guide the investigation of hypothesized relationships in data. For example, a manager may analyze

FIGURE 4-2
File-Oriented Systems Versus Database Systems



supplier purchases for the last three years. This may be followed by additional queries that "drill down" to lower levels by, for example, grouping purchases by different items and by fiscal periods. ***Data mining*** is using sophisticated statistical analysis, including artificial intelligence techniques such as neural networks, to "discover" unhypothesized relationships in the data. For example, credit card companies use data mining to identify usage patterns indicative of fraud. Similarly, data mining techniques can identify previously unknown relationships in sales data that can be used in future promotions.

Proper controls are needed to reap significant benefits from data warehousing. Data validation controls are needed to ensure that data warehouse input is accurate. Verifying the accuracy, called scrubbing the data, is often one of the most time-consuming and expensive steps in creating a data warehouse. It is also important to control access to the data warehouse as well as to encrypt the data stored in the data warehouse. Finally, it is important to regularly backup copies of the data warehouse and store them securely.

Bank of America created a customer information database to provide customer service, marketing analysis, and managerial information. It was the largest in the banking industry, with over 600 billion characters of data. It contained all bank data on checking and savings accounts; real estate, consumer, and commercial loans; ATMs; and bankcards. Although the bank spends $14 million a year to maintain the data warehouse, it is worth the cost. Queries that formerly averaged two hours took only five minutes. Minutes after Los Angeles suffered an earthquake, the bank sorted its $28 billion mortgage loan portfolio by Zip code, identified loans in the earthquake area, and calculated its potential loan loss.

The Advantages of Database Systems

Virtually all mainframes and servers use database technology, and database use in personal computers is growing rapidly. Most accountants are involved with databases through data entry, data processing, querying, or auditing. They also develop, manage, or evaluate the controls needed to ensure database integrity. Databases provide organizations with the following benefits:

- *Data integration.* Master files are combined into large "pools" of data that many application programs access. An example is an employee database that consolidates payroll, personnel, and job skills master files.
- *Data sharing.* Integrated data are more easily shared with authorized users. Databases are easily browsed to research a problem or obtain detailed information underlying a report. The FBI, which does a good job of collecting data but a poor job of sharing it, is spending eight years and $400 million to integrate data from their different systems.
- *Minimal data redundancy and data inconsistencies.* Because data items are usually stored only once, data redundancy and data inconsistencies are minimized.
- *Data independence.* Because data and the programs that use them are independent of each other, each can be changed without changing the other. This facilitates programming and simplifies data management.
- *Cross-functional analysis.* In a database system, relationships, such as the association between selling costs and promotional campaigns, can be explicitly defined and used in the preparation of management reports.

The Importance of Good Data

Incorrect database data can lead to bad decisions, embarrassment, and angry users. For example:

- A company sent half its mail-order catalogs to incorrect addresses. A manager finally investigated the large volume of returns and customer complaints. Correcting customer address in the database saved the company $12 million a year.
- Valparaiso, Indiana, used the county database to develop its tax rates. After the tax notices were mailed, a huge error was discovered: A $121,900 home was valued at $400 million and caused a $3.1 million property tax revenue shortfall. As a result, the city, the school district, and governmental agencies had to make severe budget cuts.

The Data Warehousing Institute estimates that bad data cost businesses over $600 billion a year in unnecessary postage, marketing costs, and lost customer credibility. It is estimated that over

25% of business data is inaccurate or incomplete. In a recent survey, 53% of 750 information technology (IT) professionals said their companies experienced problems due to poor-quality data.

Managing data gets harder every year: The quantity of data generated and stored doubles every 18 months. To avoid outdated, incomplete, or erroneous data, management needs policies and procedures that ensure clean, or scrubbed, data. The Sarbanes-Oxley Act states that top executives face prosecution and jail time if a company's financial data are not in order. Preventing and detecting bad data are discussed in more detail in Chapters 5 through 11.

Database Systems

Logical and Physical Views of Data

In file-oriented systems, programmers must know the physical location and layout of records. Figure 4-3 shows a ***record layout*** of an accounts receivable file. Suppose a programmer wants a report showing customer number, credit limit, and current balance. To write the program, she must understand the location and length of the fields needed (i.e., record positions 1 through 10 for customer number) and the format of each field (alphanumeric or numeric). The process becomes more complex if data from several files are used.

Database systems overcome this problem by separating the storage of the data from the use of data elements. The database approach provides two separate views of the data: the physical view and the logical view. The ***logical view*** is how people conceptually organize and understand the data. For example, a sales manager views all customer information as being stored in a table. The ***physical view*** refers to how and where data are physically arranged and stored in the computer system.

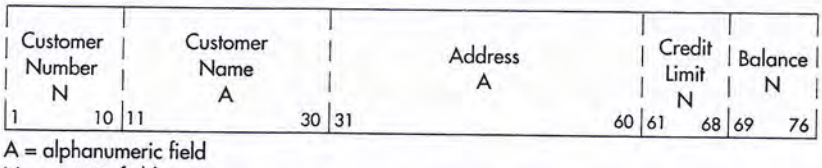
As shown in Figure 4-4, database management (DBMS) software links the way data are physically stored with each user's logical view of the data. The DBMS allows users to access, query, or update the database without reference to how or where data are physically stored. Separating the logical and physical views of data also means that users can change their logical view of data without changing the way data are physically stored. Likewise, the database administrator can change physical storage to improve system performance without affecting users or application programs.

Schemas

A ***schema*** describes the logical structure of a database. There are three levels of schemas: the conceptual, the external, and the internal. Figure 4-5 shows the relationships among these three levels. The ***conceptual-level schema***, the organization wide view of the *entire* database, lists all data elements and the relationships among them. The ***external-level schema*** consists of individual user views of portions of the database, each of which is referred to as a ***subschema***. The ***internal-level schema***, a low-level view of the database, describes how the data are stored and accessed, including record layouts, definitions, addresses, and indexes. Figure 4-5 connects each of the levels with bidirectional arrows to represent schema mappings. The DBMS uses the mappings to translate a user's or a program's request for data (expressed in terms of logical names and relationships) into the indexes and addresses needed to physically access the data.

At S&S, the conceptual schema for the revenue cycle database contains data about customers, sales, cash receipts, sales personnel, cash, and inventory. External subschemas are derived from this schema, each tailored to the needs of different users or programs. Each subschema can prevent access to those portions of the database that do not apply to it. For example,

FIGURE 4-3
Accounts Receivable File Record Layout



A = alphanumeric field
N = numeric field

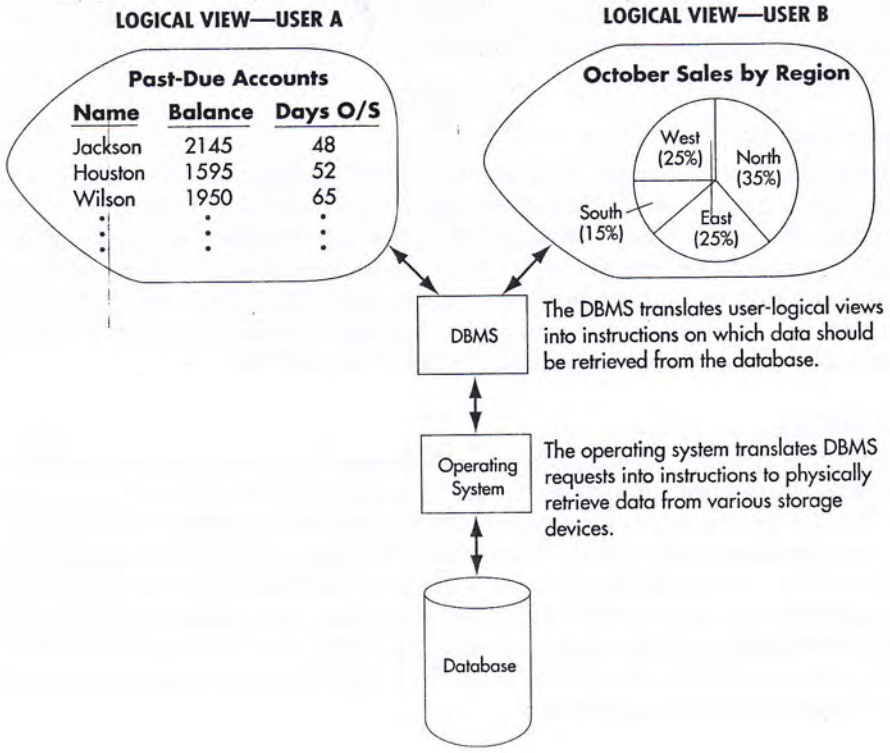


FIGURE 4-4
Function of the DBMS: To Support Multiple Logical Views of Data

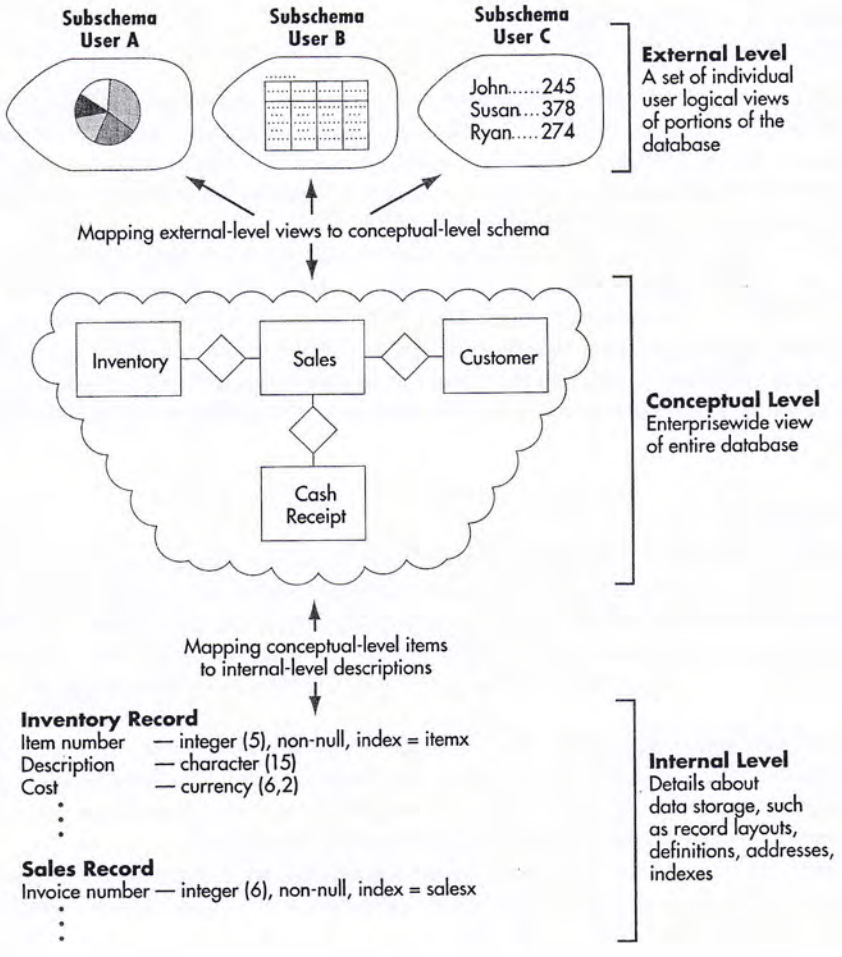


FIGURE 4-5
Three Levels of Schemas

the sales order entry subschema includes data about customer credit limits, current balances, and inventory quantities and prices. It would not include the cost of inventory or bank account balances.

The Data Dictionary

A ***data dictionary*** contains information about the structure of the database. As shown in Table 4-1, for each data element stored in the database, there is a record in the dictionary describing it. The DBMS maintains the data dictionary, whose inputs include new or deleted data elements and changes in data element names, descriptions, or uses. Outputs include reports for programmers, designers, and users such as (1) programs or reports using a data item, (2) synonyms for the data elements in a file, and (3) data elements used by a user. These reports are used for system documentation, for database design and implementation, and as part of the audit trail.

DBMS Languages

A DBMS has several languages. The ***data definition language (DDL)*** builds the data dictionary, creates the database, describes logical views for each user, and specifies records or field security constraints. The ***data manipulation language (DML)*** changes database content, including data element updates, insertions, and deletions. The ***data query language (DQL)*** contains powerful, easy-to-use commands that enable users to retrieve, sort, order, and display data. A ***report writer*** simplifies report creation. Users specify the data elements they want printed, and the report writer searches the database, extracts the data elements, and prints them in the user-specified format. The DQL and report writer are available to users. The DDL and DML should be restricted to authorized administrators and programmers.

Relational Databases

A DBMS is characterized by the logical ***data model***, or abstract representation of database contents, upon which it is based. As most new DBMSs are relational databases, this chapter focuses primarily on them. The ***relational data model*** represents conceptual- and external-level schemas as if data are stored in tables like the one shown in Table 4-2. The data are actually stored not in tables, but in the manner described in the internal-level schema.

Each row in a table, called a ***tuple*** (rhymes with *couple*), contains data about a specific occurrence of the type of entity represented by that table. Each column contains data about an attribute of that entity. For example, each row in Table 4-2 contains data about a particular inventory item that S&S carries, and each column contains data about specific inventory attributes, such as description, color, and price. Similarly, each row in a customer table contains data about a specific customer, and each column contains data about customer attributes, such as name and address.

Types of Attributes

A ***primary key*** is the database attribute, or combination of attributes, that uniquely identifies a specific row in a table. The primary key in Table 4-2 is Item Number as it uniquely identifies each merchandise item that S&S sells. Usually, the primary key is a single attribute. In some tables, two or more attributes are needed to identify uniquely a specific row in a table. The primary key of the Sales-Inventory table in Table 4-5 is the combination of Sales Invoice # and Item #.

A ***foreign key, which*** is an attribute that is a primary key in another table, is used to link tables. Customer # in Table 4-5 is the primary key in the Customer table and a foreign key in the Sales table. In the Sales table, Customer # links a sale to data about the customer who made the purchase, as contained in the Customer table (see arrows connecting tables).

Other nonkey attributes in a table store important information about that entity. The inventory table in Table 4-2 contains information about the description, color, vendor number, quantity on hand, and price of each item S&S carries.

TABLE 4-1 Example of a Data Dictionary

Data Element Name	Description	Records in Which Contained	Source	Field Length	Field Type	Programs in Which Used	Outputs in Which Contained	Authorized Users	Other Data Names
Customer number	Unique identifier of each customer	A/R record, customer record, sales analysis record	Customer number listing	10	Numeric	A/R update, customer file update, sales analysis update, credit analysis	A/R aging report, customer status report, sales analysis report, credit report	No restrictions	None
Customer name	Complete name of customer	Customer record	Initial customer order	20	Alphanumeric	Customer file update, statement processing	Customer status report, monthly statement	No restrictions	None
Address	Street, city, state, and Zip code	Customer record	Credit application	30	Alphanumeric	Customer file update, statement processing	Customer status report, monthly statement	No restrictions	None
Credit limit	Maximum credit that can be extended to customer	Customer record, A/R record	Credit application	8	Numeric	Customer file update, A/R update, credit analysis	Customer status report, A/R aging report, credit report	D. Dean R. Dalebout H. Heaton	CR_limit
Balance	Balance due from customer on credit purchases	A/R record, sales analysis record	Various sales and payment transactions	8	Numeric	A/R update, sales analysis update, statement processing, credit analysis	A/R aging report, sales analysis report, monthly statement, credit report	G. Burton B. Heninger S. Summers	Cust_bal

TABLE 4-2 Sample Inventory Table for S&S

Inventory : Table

Item Number	Description	Color	Vendor Number	Quantity on Hand	Price
1036	Refrigerator	White	10023	12	1199
1038	Refrigerator	Almond	10023	7	1299
1039	Refrigerator	Hunter Green	10023	5	1499
2061	Range	White	10011	6	799
2063	Range	Black	10011	5	999
3541	Washer	White	10008	15	499
3544	Washer	Black	10008	10	699
3785	Dryer	White	10019	12	399
3787	Dryer	Almond	10019	8	499

Record: 10 of 10

Designing a Relational Database for S&S, Inc.

In a manual accounting system, S&S would capture sales information on a preprinted sales invoice that provides both a logical and physical view of the data collected. Physical storage of sales invoice data is simple; S&S stores a copy of the invoice in a file cabinet.

Storing the same data in a computer is more complex. Suppose S&S wanted to store five sales invoices (numbered 101 to 105) electronically. On several invoices, a customer buys more than one item. Let us look at the effects of several ways of storing this information.

1: Store All Data in One Uniform Table. S&S could store sales data in one table, as illustrated in Table 4-3. This approach has two disadvantages. First, it stores lots of redundant data. Examine invoice 102 in Table 4-3. Because three inventory items are sold, invoice and customer data (columns 1–9) are recorded three times. Likewise, inventory descriptions and unit prices are repeated each time an item is sold. Because sales volumes are high in a retail store (remember, Table 4-3 represents only five invoices), such redundancy makes file maintenance unnecessarily time-consuming and error-prone.

Second, problems occur when invoice data are stored in one table. The first is called an ***update anomaly***, because data value updates are not correctly recorded. Changing a customer's address involves searching the entire table and changing every occurrence of that customer's address. Overlooking even one row creates an inconsistency, because multiple addresses would exist for the same customer. This could result in unnecessary duplicate mailings and other errors.

An ***insert anomaly*** occurs in our example because there is no way to store information about prospective customers until they make a purchase. If prospective customer data is entered before a purchase is made, the Sales Invoice # column would be blank. However, the sales invoice number is the primary key for Table 4-3 and cannot be blank, as it uniquely identifies the record.

A ***delete anomaly*** occurs when deleting a row has unintended consequences. If a customer had one purchase of a single item, deleting that row erases all data about that customer.

2: Vary the Number of Columns. An alternative to Table 4-3 is to record sales invoice and customer data once and add additional columns to record each item sold. Table 4-4 illustrates this approach. Although this reduces data redundancy and eliminates some anomalies associated with Table 4-3, it has drawbacks. S&S would have to decide in advance how many item numbers to leave room for in each row (that is, how many columns to put in the table; note in Table 4-4 that to store each additional item requires five additional columns—Item, Quantity, Description, Unit Price, and Extended Amount). If room is left for four items (20 columns), how would data about a sale involving eight items (40 columns) be stored? If room is left for eight items, however, there will be a great deal of wasted space, as is the case for sales invoices 103 and 104.

TABLE 4-3 Example of Storing All Sales Data for S&S in One Table

Sales : Table

Sales Invoice #	Date	Salesperson	Customer #	Invoice Total	Customer Name	Street	City	State	Item #	Quantity	Description	Unit Price	Extended Amount
101	10/15/2015	J. Buck	151	1447	D. Ainge	123 Lotus Lane	Phoenix	AZ	10	2	Television	499	998
101	10/15/2015	J. Buck	151	1447	D. Ainge	123 Lotus Lane	Phoenix	AZ	50	1	Microwave	449	449
102	10/15/2015	S. Knight	152	4394	G. Kite	40 Quatro Road	Mesa	AZ	10	1	Television	499	499
102	10/15/2015	S. Knight	152	4394	G. Kite	40 Quatro Road	Mesa	AZ	20	3	Freezer	699	2097
102	10/15/2015	S. Knight	152	4394	G. Kite	40 Quatro Road	Mesa	AZ	30	2	Refrigerator	899	1798
103	10/15/2015	S. Knight	151	898	D. Ainge	123 Lotus Lane	Phoenix	AZ	50	2	Microwave	449	898
104	10/15/2015	J. Buck	152	789	G. Kite	40 Quatro Road	Mesa	AZ	40	1	Range	789	789
105	11/14/2015	J. Buck	153	3994	F. Roberts	401 Excel Way	Chandler	AZ	10	3	Television	499	1497
105	11/14/2015	J. Buck	153	3994	F. Roberts	401 Excel Way	Chandler	AZ	20	1	Freezer	699	699
105	11/14/2015	J. Buck	153	3994	F. Roberts	401 Excel Way	Chandler	AZ	30	2	Refrigerator	899	1798

Record: 11 of 11

TABLE 4-4 Example of Storing S&S Sales Data by Adding Columns for Each Additional Item Sold

Sales : Table

Sales Invoice #	Columns 2-9	Item #	Quantity	Description	Unit Price	Extended Amount	Item #2	Quantity2
101	Same 8	10	2	Television	499	998	50	1
102	columns	10	1	Television	499	499	20	3
103	as in	50	2	Microwave	449	898		
104	Table 4-3	40	1	Range	789	789		
105	above	10	3	Television	499	1497	20	1

Record: 6 of 6

3. The Solution: A Set of Tables. The storage problems in Tables 4-3 and 4-4 are solved using a *relational database*. The set of tables in Table 4-5 represent a well-structured relational database.

Basic Requirements of a Relational Database

We now turn to the guidelines used to develop a properly structured relational database.

1. ***Every column in a row must be single valued.*** In a relational database, there can only be one value per cell. At S&S, each sale can involve more than one item. On invoice 102, the customer bought a television, a freezer, and a refrigerator. If Item # were an attribute in the Sales table, it would have to take on three values (item numbers 10, 20, and 30). To solve this problem, a Sales-Inventory table was created that lists each item sold on an invoice. The third line in the Sales-Inventory table in Table 4-5 shows invoice 102 and item number 10 (television). The fourth line shows invoice 102 and item 20 (freezer). The fifth line shows invoice 102 and item 30 (refrigerator). This table repeats the invoice number as often as needed to show all the items purchased on a sales invoice.

2. ***Primary keys cannot be null.*** A primary key cannot uniquely identify a row in a table if it is null (blank). A non-null primary key ensures that every row in a table represents something and that it can be identified. This is referred to as the ***entity integrity rule***. In the Sales-Inventory table in Table 4-5, no single field uniquely identifies each row. However, the first two columns, taken together, do uniquely identify each row and constitute the primary key.

3. ***Foreign keys, if not null, must have values that correspond to the value of a primary key in another table.*** Foreign keys link rows in one table to rows in another table. In Table 4-5, Customer # can link each sales transaction with the customer who participated in that event only if the Sales table Customer # value corresponds to an actual customer number in the Customer table. This constraint, called the ***referential integrity rule***, ensures database consistency. Foreign keys can contain null values. For example, when customers pay cash, Customer # in the sales table can be blank.

4. ***All nonkey attributes in a table must describe a characteristic of the object identified by the primary key.*** Most tables contain other attributes in addition to the primary and foreign keys. In the Customer table in Table 4-5, Customer # is the primary key, and customer name, street, city, and state are important facts that describe the customer.

These four constraints produce a well-structured (normalized) database in which data are consistent and redundancy is minimized and controlled. In Table 4-5, having a table for each entity of interest avoids the anomaly problems discussed previously and minimizes redundancy. Redundancy is not eliminated, as certain items, such as Sales Invoice #, appear in more than one table when they are foreign keys. The referential integrity rule ensures that there are no update anomaly problems with the foreign keys.

When data about objects of interest are stored in separate database tables, it is easy to add new data by adding another row to the table. For example, adding a new customer is as simple as adding a new row to the Customer table. Thus, the tables depicted in Table 4-5 are free from insert anomalies.

Relational databases also simplify data deletion. Deleting sales invoice 105, the only sale to customer 153, does not erase all data about that customer, because it is stored in the Customer table. This avoids delete anomalies.

Description2	Unit Price2	Extended Amou	Item #3	Quantity3	Description3	Unit Price3	Extended Amount3
Microwave	449	449					
Freezer	699	2097	30	2	Refrigerator	899	1798
Freezer	699	699	30	2	Refrigerator	899	1798
of 6							

Another benefit of the schema shown in Table 4-5 is that space is used efficiently. The Sales-Inventory table contains a row for each item sold on each invoice. There are no blank rows, yet all sales data are recorded. In contrast, the schema in Table 4-4 results in much wasted space.

Two Approaches to Database Design

One way to design a relational database, called ***normalization***, assumes that everything is initially stored in one large table. Rules are followed to decompose that initial table into a set of tables in what is called *third normal form (3NF)*, because they are free of update, insert, and delete anomalies. The details of the normalization process can be found in any database textbook.

In an alternative design approach, called ***semantic data modeling***, the designer uses knowledge of business processes and information needs to create a diagram that shows what to include in the database. This diagram is used to create a set of relational tables that are already in 3NF.

Semantic data modeling has significant advantages. First, using a system designer's knowledge of business processes facilitates the efficient design of transaction processing databases. Second, the graphical model explicitly represents the organization's business processes and policies and, by facilitating communication with system users, helps ensure that the new system meets users' actual needs. Chapter 17 introduces two semantic data modeling tools, entity-relationship diagramming and REA modeling, used to design transaction processing databases.

Creating Relational Database Queries

To retrieve stored data, users query databases. This section of the chapter shows you how to query databases using Microsoft Access. If you want to follow along by creating the queries illustrated in this section, download the S&S In-Chapter Database from the text's Web site. When you open the database and select the Create ribbon, the ribbon in the top half of Table 4-6 appears. There are two ways to query the database: create a query in Design view (the "Query Design" button) or use the wizard (the "Query Wizard" button). These options are highlighted in the top half of Table 4-6. In the example shown in Table 4-6, the Design view is used. Clicking on the "Query Design" button produces the Show Table window shown in Table 4-6. The user selects the tables needed to produce the desired information; if more tables than necessary are selected, the query may not run properly.

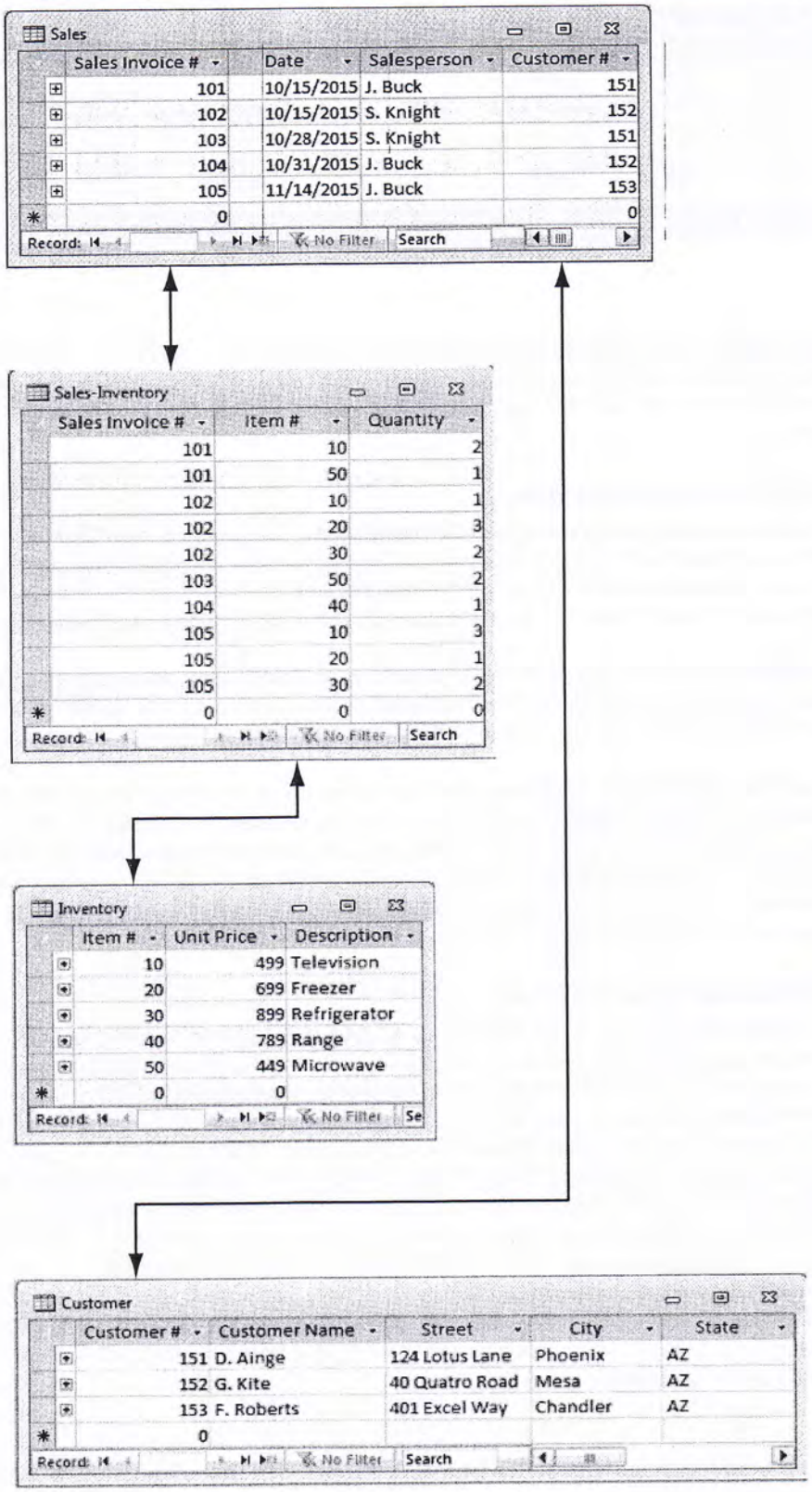
We will use the tables in Table 4-5 to walk through the steps needed to create and run five queries. This will not make you an expert in querying an Access database, but it will show you how to produce useful information.

Query 1

Query 1 answers two questions: What are the invoice numbers of all sales made to D. Ainge, and who was the salesperson for each sale?

The Sales and Customer tables contain the three items needed to answer this query: Sales Invoice #, Salesperson, and Customer Name. Click the "Query Design" button (see Table 4-6), and select the Sales and Customer tables by double-clicking on their names or by single-clicking

TABLE 4-5 Set of Relational Tables for Storing S&S Sales Data



Sales

Sales Invoice #	Date	Salesperson	Customer #
101	10/15/2015	J. Buck	151
102	10/15/2015	S. Knight	152
103	10/28/2015	S. Knight	151
104	10/31/2015	J. Buck	152
105	11/14/2015	J. Buck	153
0			0

Sales-Inventory

Sales Invoice #	Item #	Quantity
101	10	2
101	50	1
102	10	1
102	20	3
102	30	2
103	50	2
104	40	1
105	10	3
105	20	1
105	30	2
0	0	0

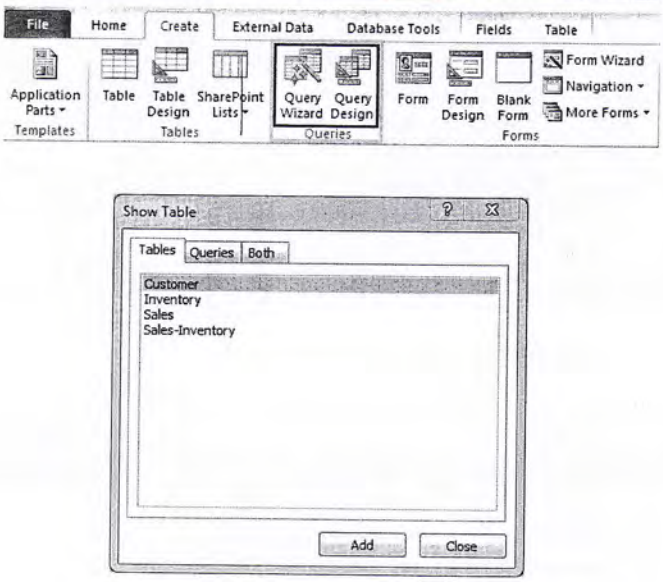
Inventory

Item #	Unit Price	Description
10	499	Television
20	699	Freezer
30	899	Refrigerator
40	789	Range
50	449	Microwave
0	0	

Customer

Customer #	Customer Name	Street	City	State
151	D. Ainge	124 Lotus Lane	Phoenix	AZ
152	G. Kite	40 Quatro Road	Mesa	AZ
153	F. Roberts	401 Excel Way	Chandler	AZ
0				

TABLE 4-6 Creating Queries in the Microsoft Access Database



on the name and clicking the Add button. The selected tables appear as shown in Table 4-7. A line between the two tables connects the Customer # fields (the Customer table primary key and the Sales table foreign key). Click on Close to make the Show Table window disappears.

To populate the bottom half of the screen shown in Table 4-7, double-click on Sales Invoice #, Salesperson, and Customer Name or drag and drop them into the Field row. Access automatically checks the box in the Show line, so the item will be shown when the query is run.

Since we only want sales to D. Ainge, enter that in the criteria line of the Customer Name column. Run the query by clicking on the red ! (exclamation) mark on the Query Tools Design ribbon. Table 4-8 shows the tables used, the relationship of the primary and foreign keys between tables, and the query answer. The query answer does not automatically have the title "Ainge Sales." To assign the query a name, save it by selecting File from the Access menu, then Save Object As, and then enter "Ainge Sales" in the first line of the Save As window, making sure the Object select box is set to "Query," and then clicking OK. When the query is rerun, the title shown in Table 4-8 will appear.

TABLE 4-7 Completed Query 1

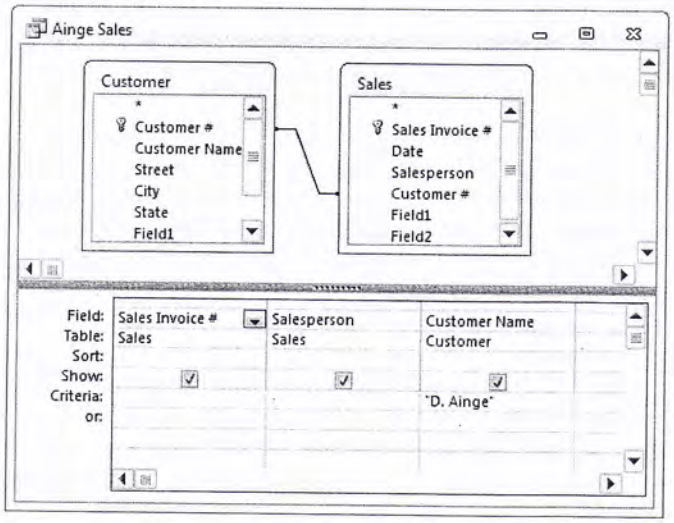
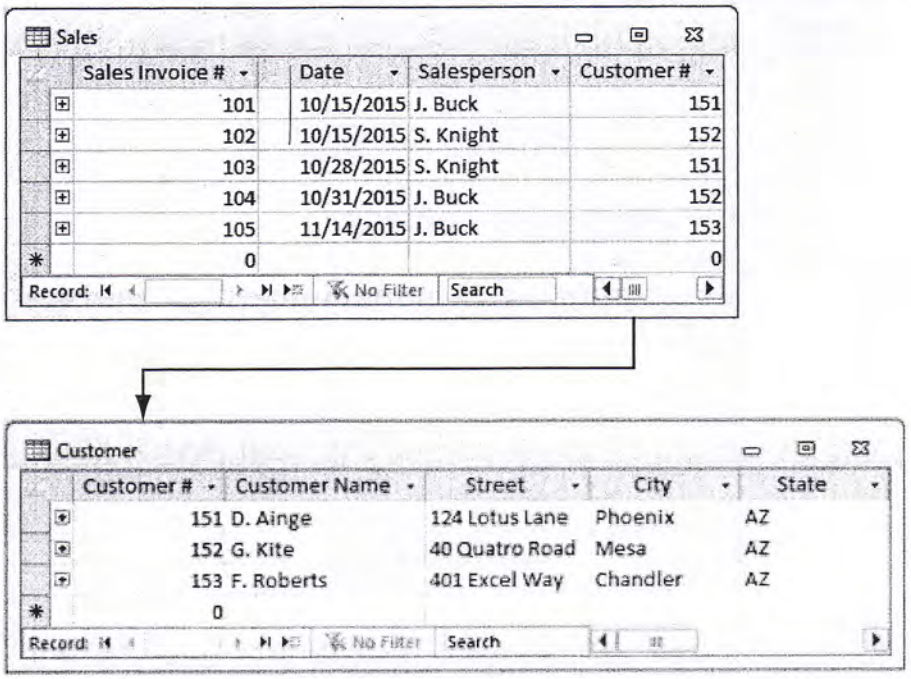


TABLE 4-8 Query 1 Relationships and Query Answer



Sales

Sales Invoice #	Date	Salesperson	Customer #
101	10/15/2015	J. Buck	151
102	10/15/2015	S. Knight	152
103	10/28/2015	S. Knight	151
104	10/31/2015	J. Buck	152
105	11/14/2015	J. Buck	153
0			0

Record: 1 of 5 No Filter Search

Customer

Customer #	Customer Name	Street	City	State
151	D. Ainge	124 Lotus Lane	Phoenix	AZ
152	G. Kite	40 Quatro Road	Mesa	AZ
153	F. Roberts	401 Excel Way	Chandler	AZ
0				

Record: 1 of 3 No Filter Search

Query Answer

Ainge Sales

Sales Invoice #	Salesperson	Customer Name
101	J. Buck	D. Ainge
103	S. Knight	D. Ainge

Record: 1 of 2 No Filter Search

Query 2

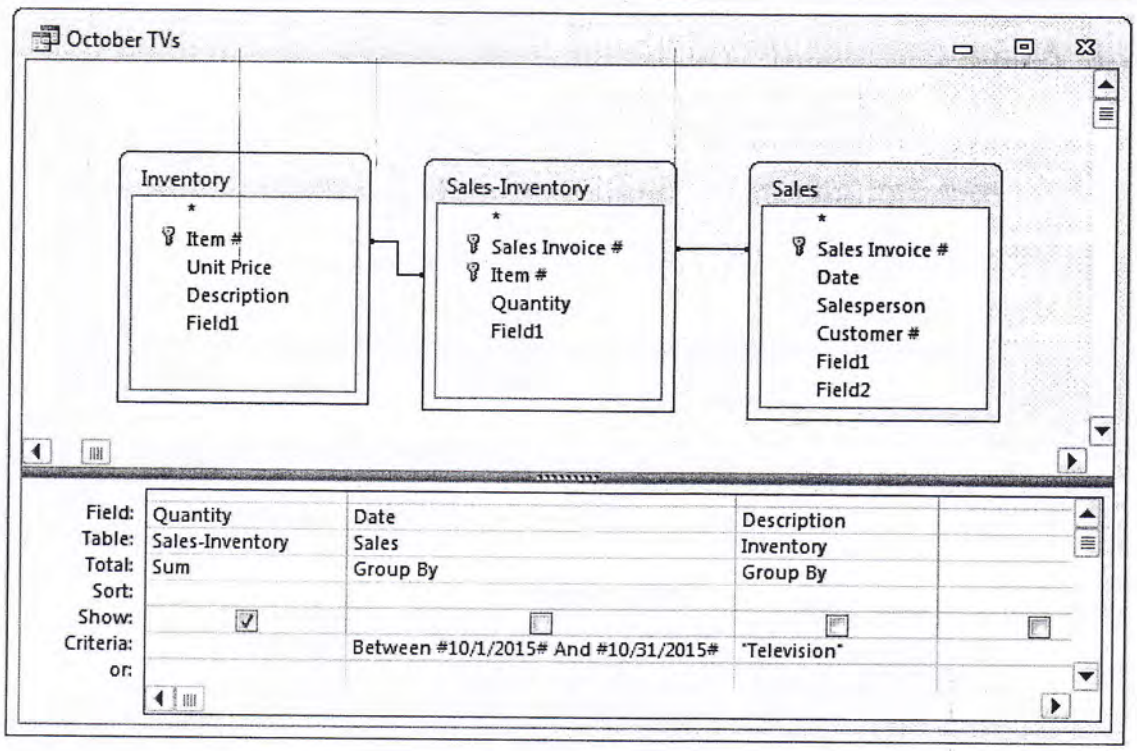
Query 2 answers this question: How many televisions were sold in October?

The Sales, Inventory, and Sales-Inventory tables contain the three items needed to answer this query: Date, Inventory Description, and Quantity. Click on the "Query Design" button in the Create ribbon and select the three tables and the three fields, as shown in Table 4-9. Since we want the quantity of televisions sold in October, we add the criteria "Between #10/1/2015# And #10/31/2015#" to the Date field and "Television" to the Description field.

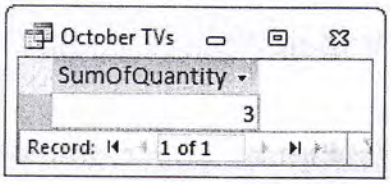
To specify criteria, Access uses operators such as "And" "Or" and "Between." An "And" operator returns the data that meets *all* the criteria linked by "And" operators. The "Between" operator selects all the data in October of 2015; that is, between and including the first and last days of the month. If the "Between" operator was not used and "Or" replaced "And," all televisions sold either on 10/1/2015 or on 10/31/2015 would be selected. The "#" symbol tells Access to look for a date rather than some other type of text.

Since we are only looking for total television sales in October, we uncheck the "Show" box in the Date and Description columns. To generate total sales, click the "Totals" button in the Show/Hide portion of the Query Tools Design ribbon. A new line, called Total, appears (compare Tables 4-7 and 4-9). Click on the Totals line in the Quantity column, then click on the down-arrow symbol, and select Sum from the drop-down menu that appears. The remaining two fields in the Total line will stay as Group By. Running the query in Table 4-9 produces the answer shown.

TABLE 4-9 Completed Query 2 and Answer



Query Answer



Query 3

Query 3 answers this question: What are the names and addresses of customers buying televisions in October?

This query needs these fields: Date (to select October sales), Description (to select televisions), and Customer Name, Street, City, and State (the information requested). All four tables are used because the Sales-Inventory table is used to move between the Sales and Inventory tables. The query uses the same criteria as Query 2. The date and description data do not need to be displayed, so the boxes in the Show line are unchecked. Running the query produces the answer shown in Table 4-10.

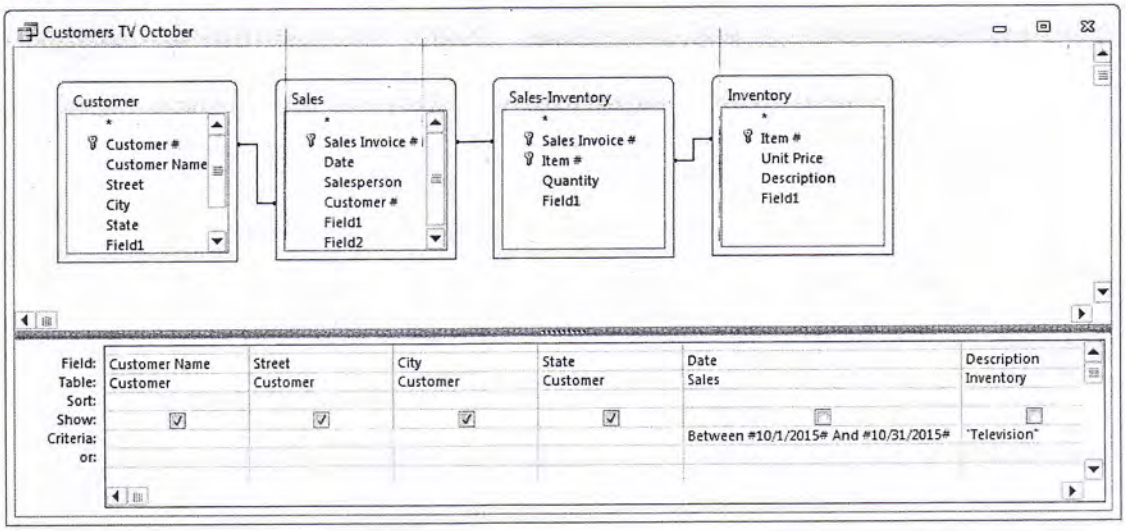
Query 4

Query 4 answers this question: What are the sales invoice numbers, dates, and invoice totals for October sales, arranged in descending order by sale amount?

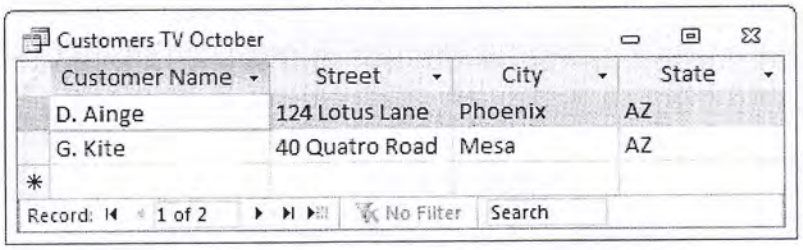
Since the database does not contain an Invoice Total column, it is calculated as follows:

1. For each invoice, calculate the total sales price of each item sold by multiplying the Quantity field in the Sales-Inventory table by the Unit Price field in the Inventory table. The Sales-Inventory table in Table 4-5 shows that three items were sold on Sales Invoice 102. For item 20, we multiply the quantity (3) by the Unit Price (699), producing 2,097. The same calculation is made for items 10 and 30.
2. Sum the three item totals to get an invoice total.

TABLE 4-10 Completed Query 3 and Answer



Query Answer

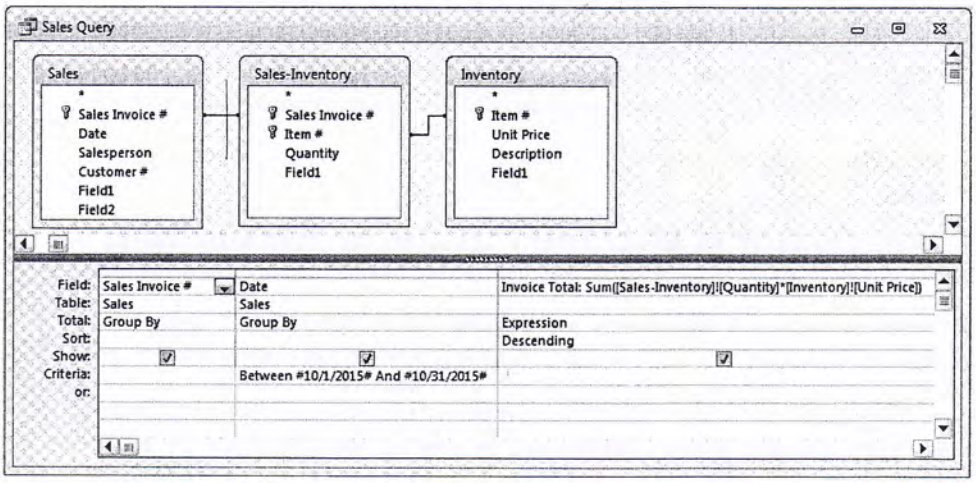


Query 4 requires the Sales table (Date, Sales Invoice #), Sales-Inventory table (Quantity), and the Inventory table (Unit Price). However, some fields will not appear in columns on the Select Query window. As shown in Table 4-11, three columns are displayed: Sales Invoice #, Date, and Invoice Total, which we will calculate. The other fields are used in the Invoice Total calculations.

To calculate Invoice Total, type "Invoice Total:" in the first blank Field cell, right-click in the cell, and select Build from the pop-up menu that appears. An Expression Builder window (see Table 4-12) appears, where the formula to calculate the invoice total is entered by typing "Sum( )". Between the parentheses, click on the + sign in front of the S&S In-Chapter Database folder in the Expressions Elements box. Then clicking on the + sign in the Tables folder causes the four database tables to appear. Click on the Sales-Inventory table, and the fields in the Sales-Inventory table appear. Double-click on Quantity to put this field in the expression. Note in Table 4-12 that the expression shows the table name and the field name, separated by an exclamation point. To multiply Quantity by Unit Price, type * (the multiplication symbol) and select the Inventory table and the Unit Price field. The formula is now complete, and the screen will appear as shown. To enter the expression into the Select Query window, click on OK.

To complete Query 4, click the Totals button in the Query Tools Design ribbon. Click on the down arrow in the Total row of the Invoice Totals column, and select Expression from the pop-up menu. This tells Access to calculate the indicated expression for all items with the same sales invoice number and date. In the same column, click on the down arrow in the Sort row, and select Descending so that the answer is show in descending Invoice Total order. In the criteria section of the Date column, use the "Between" operator to specify the month of October. Running Query 4 produces the answer shown in Table 4-11.

TABLE 4-11 Completed Query 4 and Answer



Query Answer

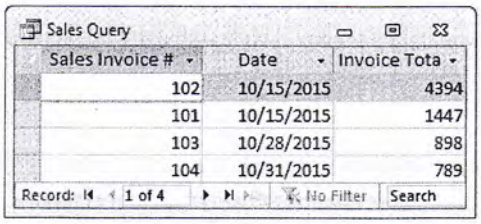
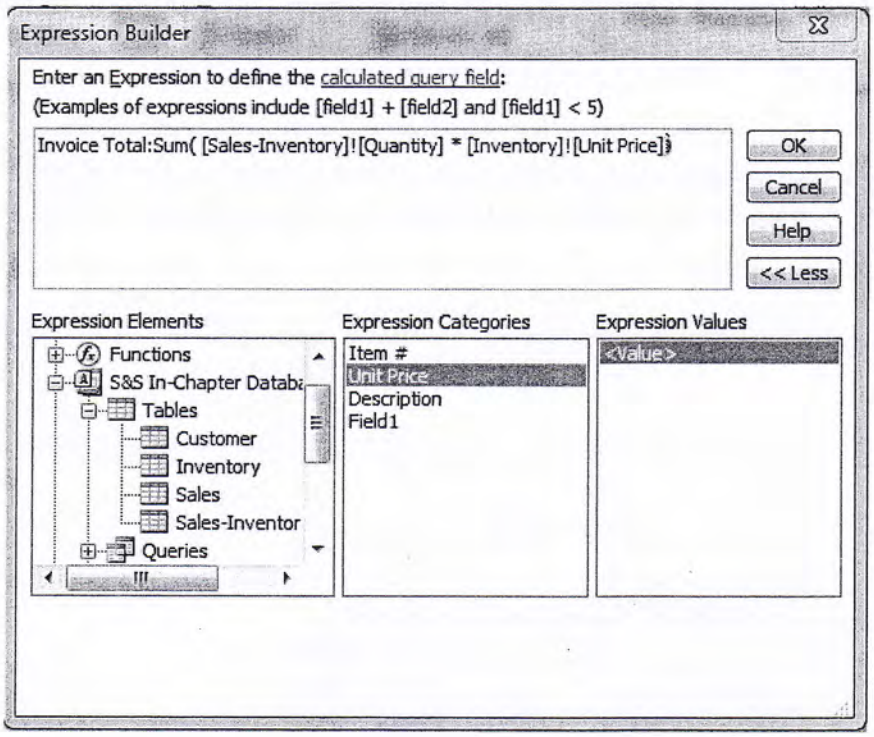


TABLE 4-12 Expression Builder for Query 4



Query 5

Query 5 will answer this question: What are total sales by salesperson?

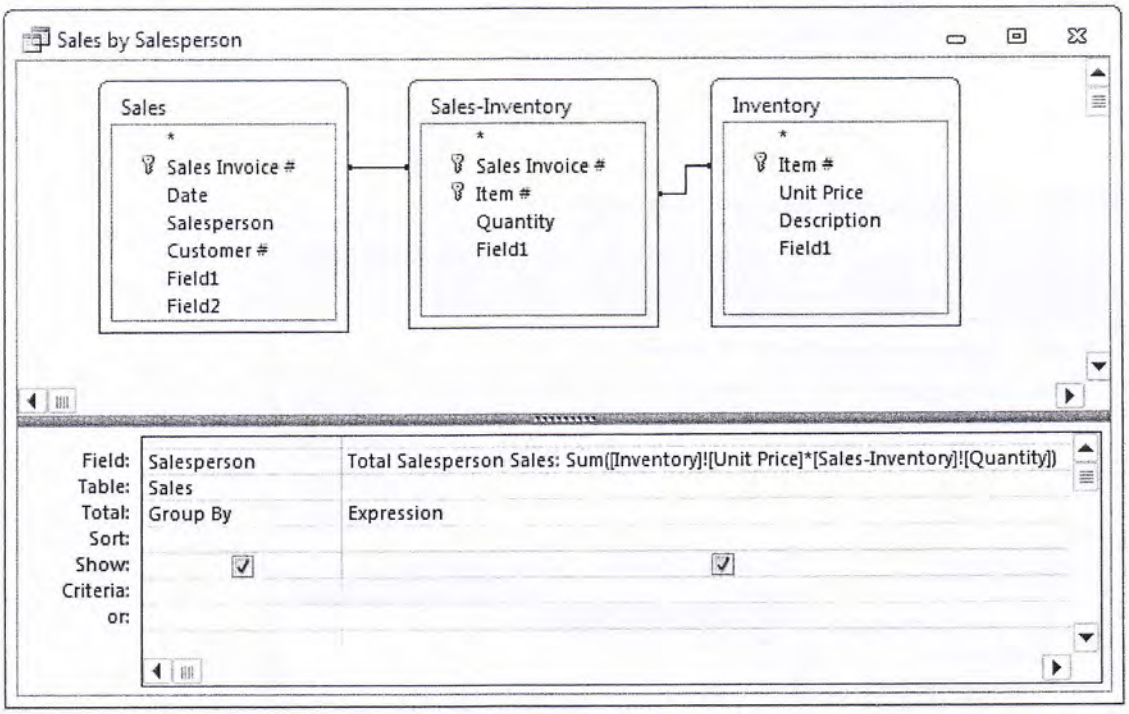
This query is similar to Query 4, except that we total invoices by salesperson rather than by invoice number. We are also not confining our query to the month of October. Try coming up with the query by yourself. The completed query and the answer are shown in Table 4-13.

Database Systems and the Future of Accounting

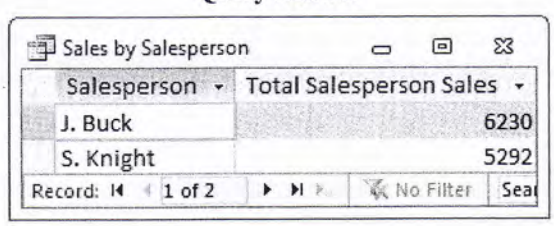
Database systems have the potential to alter external reporting significantly. Considerable time and effort are currently invested in defining how companies should summarize and report accounting information to external users. In the future, companies may make a copy of the company's financial database available to external users in lieu of the traditional financial statements. Users would be free to analyze the raw data however they see fit. Focus 4-1 discusses this possibility in more detail.

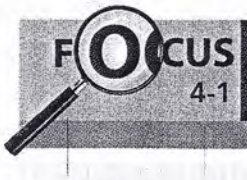
A significant advantage of database systems is the ability to create ad hoc queries to provide the information needed for decision making. No longer is financial information available only in predefined formats and at specified times. Instead, powerful and easy-to-use relational

TABLE 4-13 Query 5



Query Answer





Databases or Financial Statements?

Although information technology has dramatically changed the way business is conducted, it has had less effect on external reporting. Companies still produce highly aggregated, periodic (quarterly and annual) financial reports of past activities based on historical costs. It is estimated that the average financial database of a large company is over 100 gigabytes. Yet annual reports contain only 100 kilobytes of data. Consequently, users see only a small portion of company data presented in a predefined financial statement format.

Why not replace the annual report with a copy of the company's financial database? Many companies already give suppliers and customers limited access to their databases. Suppliers are given access to inventory data to plan production and deliveries. However, they are not given access to payroll data. Companies could similarly exclude data too sensitive to fall into competitors' hands. This database view could be placed on the Internet for investors, creditors, and other external users. The computing power of PCs makes processing such a database feasible, and external users would get a fuller and timelier picture of the organization's performance.

In such a system, the company's primary financial reporting function would be defining data elements and database structure. Users would be free to aggregate and classify that information using whatever decision model they believed to be appropriate.

Source: Robert K. Elliot, "Confronting the Future: Choices for the Attest Function," *Accounting Horizons* (September 1994): 106–124.

database query languages can find and prepare the information management needs, when they want it.

Relational DBMSs can also accommodate multiple views of the same underlying phenomenon. For example, tables storing information about assets can include columns not only for historical costs but also for current replacement costs and market values. Thus, managers will no longer be forced to look at data in ways predefined by accountants.

Finally, relational DBMSs are capable of integrating financial and operational data. For example, customer satisfaction data could be stored in the database, giving managers a richer set of data for decision making.

Relational DBMSs have the potential to increase the use and value of accounting information. Accountants must understand database systems so they can help design and use the AISs of the future. Such participation is important for ensuring that adequate controls are included in those systems to safeguard the data and ensure the reliability of the information produced.

Summary and Case Conclusion

Ashton prepared a report for Scott and Susan summarizing what he knew about databases. He explained that a database management system (DBMS), the software that makes a database system work, is based on a logical data model that shows how users perceive the way the data are stored. Many DBMSs are based on the relational data model that represents data as being stored in tables. Every row in a relational table has only one data value in each column. Neither row nor column position is significant. These properties support the use of simple, yet powerful, query languages for interacting with the database. Users only need to specify the data they want and do not need to be concerned with how the data are retrieved. The DBMS functions as an intermediary between the user and the database, thereby hiding the complex addressing schemes actually used to retrieve and update the information stored in the database.

After reading Ashton's report, Scott and Susan agreed that it was time to upgrade S&S's AIS and to hire a consulting firm to help select and install the new system. They asked Ashton to oversee the design process to ensure that the new system meets their needs.

Key Terms

database 107
database management system (DBMS) 108
database system 108
database administrator (DBA) 108
data warehouse 108
business intelligence 108
online analytical processing (OLAP) 108
data mining 109
record layout 110
logical view 110
physical view 110
schema 110
conceptual-level schema 110
external-level schema 110
subschema 110
internal-level schema 110
data dictionary 112
data definition language (DDL) 112
data manipulation language (DML) 112
data query language (DQL) 112
report writer 112
data model 112
relational data model 112
tuple 112
primary key 112
foreign key 112
update anomaly 114
insert anomaly 114
delete anomaly 114
relational database 116
entity integrity rule 116
referential integrity rule 116
normalization 117
semantic data modeling 117

AIS IN ACTION

Chapter Quiz

1. The relational data model portrays data as being stored in _____.
   a. hierarchies
   b. tables
   c. objects
   d. files

2. How a user conceptually organizes and understands data is referred to as the _____.
   a. physical view
   b. logical view
   c. data model view
   d. data organization view

3. What is each row in a relational database table called?
   a. relation
   b. attribute
   c. anomaly
   d. tuple

4. Which of the following is an individual user's view of the database?
   a. conceptual-level schema
   b. external-level schema
   c. internal-level schema
   d. logical-level schema

5. Which of the following would managers most likely use to retrieve information about sales during the month of October?
   a. DML
   b. DSL
   c. DDL
   d. DQL

6. Which of the following attributes would most likely be a primary key?
   a. supplier name
   b. supplier number
   c. supplier Zip code
   d. supplier account balance

7. Which of the following is a software program that runs a database system?
   a. DQL
   b. DBMS
   c. DML
   d. DDL

8. The constraint that all primary keys must have non-null data values is referred to as which of the following?
   a. referential integrity rule
   b. entity integrity rule
   c. normalization rule
   d. relational data model rule

9. The constraint that all foreign keys must have either null values or the value of a primary key in another table is referred to as which of the following?
   a. referential integrity rule
   b. entity integrity rule
   c. foreign key value rule
   d. null value rule

10. Which of the following attributes in the Cash Receipts table (representing payments received from customers) would most likely be a foreign key?
   a. cash receipt number
   b. customer check number
   c. customer number
   d. cash receipt date

Comprehensive Problem

The Butler Financing Company runs a mortgage brokerage business that matches lenders and borrowers. Table 4-14 lists some of the data that Butler maintains on its borrowers and lenders. The data are stored in a spreadsheet that must be manually updated for each new

TABLE 4-14 Butler Financing Company Spreadsheet

Borrower Number	Last Name	First Name	Current Address	Requested Mortgage Amount	Lender Number	Lender Name	Lender Office Address	Property Appraiser Number	Property Appraiser Name
450	Adams	Jennifer	450 Peachtree Rd.	$245,000	13	Excel Mortgage	6890 Sheridan Dr.	8	Advent Appraisers
451	Adamson	David	500 Loop Highway	$124,688	13	Excel Mortgage	6890 Sheridan Dr.	9	Independent Appraisal Service
452	Bronson	Paul	312 Mountain View Dr.	$345,000	14	CCY	28 Buckhead Way	10	Jones Property Appraisers
453	Brown	Marietta	310 Loop Highway	$ 57,090	15	Advantage Lenders	3345 Lake Shore Dr.	10	Jones Property Appraisers
454	Charles	Kenneth	3 Commons Blvd.	$ 34,000	16	Capital Savings	8890 Coral Blvd.	8	Advent Appraisers
455	Coulter	Tracey	1367 Peachtree Rd.	$216,505	13	Excel Mortgage	6890 Sheridan Dr.	8	Advent Appraisers
456	Foster	Harold	678 Loop Highway	$117,090	12	National Mortgage	750 16 St.	9	Independent Appraisal Service
457	Frank	Vernon	210 Bicayne Blvd.	$ 89,000	12	National Mortgage	750 16 St.	10	Jones Property Appraisers
458	Holmes	Heather	1121 Bicayne Blvd.	$459,010	16	Capital Savings	8890 Coral Blvd.	10	Jones Property Appraisers
459	Johanson	Sandy	817 Mountain View Dr.	$ 67,900	15	Advantage Lenders	3345 Lake Shore Dr.	9	Independent Appraisal Service
460	Johnson	James	985 Loop Highway	$ 12,000	12	National Mortgage	750 16 St.	10	Jones Property Appraisers
461	Jones	Holly	1650 Washington Blvd.	$ 67,890	15	Advantage Lenders	3345 Lake Shore Dr.	9	Independent Appraisal Service

borrower, lender, or mortgage. This updating is error-prone, which has harmed the business. In addition, the spreadsheet has to be sorted in many different ways to retrieve the necessary data.

Create a database from Butler's spreadsheet that does not have any of the data anomalies explained in the chapter. To test the database, prepare a query to show which borrowers (both borrower number and name) took out loans from Excel Mortgage and who the appraiser was for each loan.

Discussion Questions

4.1. Contrast the logical and the physical views of data, and discuss why separate views are necessary in database applications. Describe which perspective is most useful for each of the following employees: a programmer, a manager, and an internal auditor. How will understanding logical data structures assist you when designing and using database systems?

4.2. The relational data model represents data as being stored in tables. Spreadsheets are another tool that accountants use to employ a tabular representation of data. What are some similarities and differences in the way these tools use tables? How might an accountant's familiarity with the tabular representation of spreadsheets facilitate or hinder learning how to use a relational DBMS?

4.3. Some people believe database technology may eliminate the need for double-entry accounting. This creates three possibilities: (1) the double-entry model will be abandoned; (2) the double-entry model will not be used directly, but an external-level schema based on the double-entry model will be defined for accountants' use; or (3) the double-entry model will be retained in database systems. Which alternative do you think is most likely to occur? Why?

4.4. Relational DBMS query languages provide easy access to information about the organization's activities. Does this mean that online, real-time processing should be used for all transactions? Does an organization need real-time financial reports? Why or why not?

4.5. Why is it so important to have good data?

4.6. What is a data dictionary, what does it contain, and how is it used?

4.7. Compare and contrast the file-oriented approach and the database approach. Explain the main advantages of database systems.

Problems

4.1. The following data elements comprise the conceptual-level schema for a database:
billing address
cost
credit limit
customer name
customer number
description
invoice number

item number
price
quantity on hand
quantity sold
shipping address
terms

Required

a. Identify three potential users and design a subschema for each. Justify your design by explaining why each user needs access to the subschema data elements.
b. Use Microsoft Access or some other relational database product to create the schema tables. Specify the primary key(s), foreign key(s), and other data for each table. Test *your model by entering sample data in each table.*

4.2. Most DBMS packages contain data definition, data manipulation, and data query languages. For each of the following, indicate which language would be used and why.
a. A database administrator defines the logical structure of the database.
b. The controller requests a cost accounting report containing a list of all employees being paid for more than 10 hours of overtime in a given week.
c. A programmer develops a program to update the fixed-assets records stored in the database.
d. The human resources manager requests a report noting all employees who are retiring within five years.
e. The inventory serial number field is extended in the inventory records to allow for recognition of additional inventory items with serial numbers containing more than 10 digits.
f. A user develops a program to print out all purchases made during the past two weeks.
g. An additional field is added to the fixed-asset records to record the estimated salvage value of each asset.

4.3. Ashton wants to store the following data about S&S's purchases of inventory:

item number
date of purchase
vendor number
vendor address
vendor name
purchase price
quantity purchased
employee number
employee name
purchase order number
description
quantity on hand
extended amount
total amount of purchase

Required

a. Design a set of relational tables to store this data. Do all of the data items need to be stored in a table? If not, which ones do not need to be stored, and why do they not need to be stored?
b. Identify the primary key for each table.
c. Identify the foreign keys needed in the tables to implement referential integrity.
d. Implement your tables using any relational database product to which you have access.
e. Test your specification by entering sample data in each table.
f. Create a few queries to retrieve or analyze the data you stored.

4.4. Retrieve the S&S In-Chapter Database (in Microsoft Access format) from the text's Web site (or create the tables in Table 4-5 in a relational DBMS product). Write queries to answer the following questions. *Note*: For some questions, you may have to create two queries—one to calculate an invoice total and the second to answer the question asked.

a. How many different kinds of inventory items does S&S sell?
b. How many sales were made during October?
c. What were total sales in October?
d. What was the average amount of a sales transaction?
e. Which salesperson made the largest sale?
f. How many units of each product were sold?
g. Which product was sold most frequently?

4.5. Enter the tables in Table 4-15 into a relational DBMS package. Write queries to answer the following questions. *Note*: For some questions, you may have to create two queries—one to calculate a total and the second to answer the question asked.

a. Which customers (show their names) made purchases from Martinez?
b. Who has the largest credit limit?
c. How many sales were made in October?
d. What were the item numbers, price, and quantity of each item sold on invoice number 103?
e. How much did each salesperson sell?
f. How many customers live in Arizona?
g. How much credit does each customer still have available?
h. How much of each item was sold? (Include the description of each item in your answer.)
i. Which customers still have more than $1,000 in available credit?
j. For which items are there at least 100 units on hand?

TABLE 4-15

Customer

Customer #	Customer Name	City	State	Credit Limit
1000	Smith	Phoenix	AZ	2500
1001	Jones	St. Louis	MO	1500
1002	Jeffries	Atlanta	GA	4000
1003	Gilkey	Phoenix	AZ	5000
1004	Lankford	Phoenix	AZ	2000
1005	Zeile	Chicago	IL	2000
1006	Pagnozzi	Salt Lake City	UT	3000
1007	Arocha	Chicago	IL	1000
0				

Inventory

Item #	Description	Unit Cost	Unit Price	Quantity on Hand
1010	Blender	$14.00	$29.95	200
1015	Toaster	$12.00	$19.95	300
1020	Mixer	$23.00	$33.95	250
1025	Television	$499.00	$699.95	74
1030	Freezer	$799.00	$999.95	32
1035	Refrigerator	$699.00	$849.95	25
1040	Radio	$45.00	$79.95	100
1045	Clock	$79.00	$99.95	300
0			$0.00	

Sales

Invoice Number	Date	Salesperson	Customer Nu	Amount
101	10/3/2015	Wilson	1000	$1,549.90
102	10/5/2015	Mahomet	1003	$299.95
103	10/5/2015	Jackson	1002	$1,449.80
104	10/15/2015	Drezen	1000	$799.90
105	10/15/2015	Martinez	1005	$849.95
106	10/16/2015	Martinez	1007	$99.95
107	10/29/2015	Mahomet	1002	$2,209.70
108	11/3/2015	Martinez	1000	$779.90
0			0	

Sales-Inventory

Invoice Number	Item Numbe	Quantity	Extension
101	1025	1	$699.95
101	1035	1	$849.95
102	1045	3	$299.85
103	1010	1	$29.95
103	1015	1	$19.95
103	1025	2	$1,399.90
104	1025	1	$699.95
104	1045	1	$99.95
105	1035	1	$849.95
106	1045	1	$99.95
107	1030	1	$999.95
107	1035	1	$849.95
107	1040	2	$159.90
107	1045	2	$199.90
108	1025	1	$699.95
108	1045	1	$99.95
0	0	0	

4.6. The BusyB Company wants to store data about its employee skills. Each employee may possess one or more specific skills, and several employees may have the same skill. Include the following facts in the database:
date hired
date of birth
date skill acquired
employee name
employee number
pay rate
skill name

TABLE 4-16 Database That Needs to Be Extended

Inventory

Item Number	Description	Quantity on Hand	List Price
10573	19" Monitor	13	$495.00
10574	21" Monitor	8	$949.00
10622	Laser Printer	22	$395.00
10623	Color Laser Printer	5	$699.00
10624	Multi-functional Printer	12	$799.00
0			

Record: 1 of 5 | No Filter | Search

Inventory Table (Item # is primary key)

Customer

Customer Number	Name	Street	City	State	Zip Code	Credit Limit	Account Balance
11255	G. Hwang	2993 Main	Mesa	AZ	85281	$4,000.00	$875.00
12971	J. Jackson	466 W. Oak	Tempe	AZ	85286	$5,000.00	$2,588.00
13629	P. Szabo	246 E. Palm	Mesa	AZ	85281	$6,000.00	$3,955.00
15637	S. Martinez	2866 Spring	Tempe	AZ	85287	$5,000.00	$250.00
18229	B. Adams	1744 Apache	Tempe	AZ	85287	$3,000.00	$1,675.00
0							

Record: 1 of 5 | No Filter | Search

Customer Table (Customer # is primary key)

Sales

Invoice Number	Date	Salesperson #	Customer #	Amount
10001	9/8/2015	25	15637	$399.00
10002	9/10/2015	22	12971	$1,748.00
10003	9/25/2015	24	13629	$1,185.00
10004	10/2/2015	25	11255	$399.00
10005	10/11/2015	22	15637	$1,098.00
10006	10/25/2015	25	18229	$990.00
0			0	

Record: 1 of 6 | No Filter | Search

Sales Table (Invoice # is primary key)

Sales-Inventory

Invoice Number	Item Number	Quantity	Actual Unit Price
10001	10573	1	$495.00
10002	10574	1	$949.00
10002	10624	1	$799.00
10003	10622	3	$395.00
10004	10573	1	$495.00
10005	10573	1	$495.00
10005	10623	1	$699.00
10006	10573	2	$495.00
0	0	0	

Record: 1 of 9 | No Filter | Search

Sales-Inventory Line Items Table (Combination of Invoice # and Item # forms primary key)

skill number
supervisor

Required

a. Design a set of relational tables to store these data.
b. Identify the primary key for each table, and identify any needed foreign keys.
c. Implement your schema using any relational DBMS. Specify primary and foreign keys, and enforce referential integrity. Demonstrate the soundness of your design by entering sample data in each table.

4.7. You want to extend the schema shown in Table 4-16 to include information about customer payments. Some customers make installment payments on each invoice. Others write a check to pay for several different invoices. You want to store the following information:
amount applied to a specific invoice
cash receipt number
customer name
customer number
date of receipt
employee processing payment
invoice payment applies to
total amount received

Required

a. Modify the set of tables in Table 4-16 to store this additional data.
b. Identify the primary key for each new table you create.
c. Implement your schema using any relational DBMS package. Indicate which attributes are primary and foreign keys, and enter sample data in each table you create.

4.8. Create relational tables that solve the update, insert, and delete anomalies in Table 4-17.

TABLE 4-17 Invoice Table

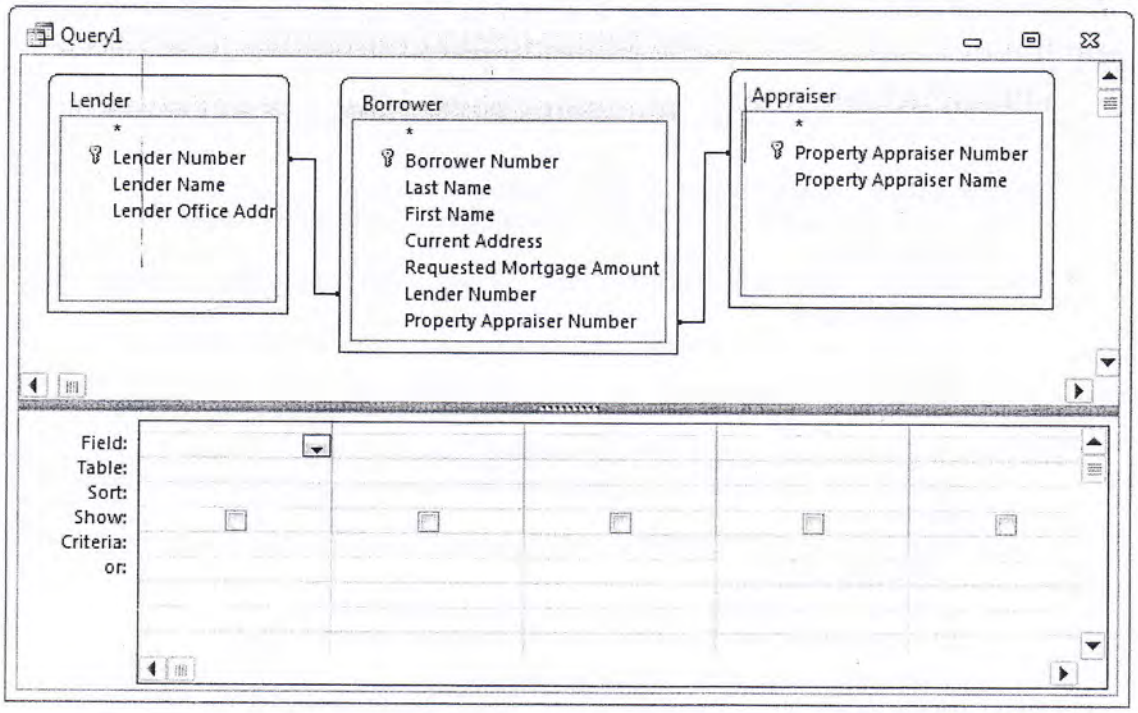
Invoice #	Date	Order Date	Customer ID	Customer Name	Item #	Description	Quantity
52	6-19-2015	5-25-2015	201	Johnson	103	Trek 9000	5
52	6-19-2015	5-25-2015	201	Johnson	122	Nimbus 4000	8
52	6-19-2015	5-25-2015	201	Johnson	10	Izzod 3000	11
52	6-19-2015	5-25-2015	201	Johnson	71	LD Trainer	12
57	6-20-2015	6-01-2015	305	Henry	535	TR Standard	18
57	6-20-2015	6-01-2015	305	Henry	115	NT 2000	15
57	6-20-2015	6-01-2015	305	Henry	122	Nimbus 4000	5

4.9. Create relational tables that solve the update, insert, and delete anomalies in Table 4-18.

TABLE 4-18 Purchase Order (PO) Table

Purchase Order #	Purchase Date	Order Part #	Description	Unit Price	Quantity Ordered	Vendor #	Vendor Name	Vendor Address
2	3/9/2015	334	XYZ	$30	3	504	KL Supply	75 Stevens Dr.
2	3/9/2015	231	PDQ	$50	5	504	KL Supply	75 Stevens Dr.
2	3/9/2015	444	YYM	$80	6	504	KL Supply	75 Stevens Dr.
3	4/5/2015	231	PDQ	$50	2	889	Oscan Inc	55 Cougar Cir.

TABLE 4-19 Selected Query Screen for Chapter Comprehensive Problem



4.10. From the database created in the comprehensive problem, perform queries based on the tables and query grid shown in Table 4-19.

a. Which borrowers use Advent Appraisers?
b. What is the average amount borrowed from National Mortgage?
c. List all of the property appraisers.
d. List all of the lenders.
e. List the lenders that lent more than $100,000.
f. Which borrower requested the largest mortgage?
g. Which borrower requested the smallest mortgage?

Case 4-1 Research Project

As in all areas of information technology, DBMSs are constantly changing and improving. Research how businesses are using DBMSs, and write a report of your findings. Address the following issues:

1. Which popular DBMS products are based on the relational data model?
2. Which DBMS products are based on a logical model other than the relational data model?
3. What are the relative strengths and weaknesses of the different types (relational versus other logical models) of DBMSs?

AIS IN ACTION SOLUTIONS

Quiz Key

1. The relational data model portrays data as being stored in _____.
   a. hierarchies (Incorrect. A hierarchical database portrays data as being stored in hierarchies.)
   ▶ b. tables (Correct. The relational data model portrays data as being stored in a table or relation format.)
   c. objects (Incorrect. An object-oriented database portrays data as being stored as objects.)
   d. files (Incorrect. The file-based data model portrays data as being stored in files.)

2. How a user conceptually organizes and understands data is referred to as the _____.
   a. physical view (Incorrect. The physical view shows how and where data are physically stored.)
   ▶ b. logical view (Correct. The logical view shows how a user conceptually organizes and *understands data.)*
   c. data model view (Incorrect. This is not a typical database view.)
   d. data organization view (Incorrect. This is not a typical database view.)

3. What is each row in a relational database table called?
   a. relation (Incorrect. A relation is a table in a relational database.)
   b. attribute (Incorrect. Each column in a relational database is an attribute that describes some characteristic of the entity about which data are stored.)
   c. anomaly (Incorrect. An anomaly is a problem in a database, such as an insert anomaly or a delete anomaly.)
   ▶ d. tuple (Correct. A tuple is also called a row in a relational database.)

4. Which of the following is an individual user's view of the database?
   a. conceptual-level schema (Incorrect. A conceptual-level schema is the organizationwide view of the entire database.)
   ▶ b. external-level schema (Correct. The external-level schema represents an individual user's view of the database.)
   c. internal-level schema (Incorrect. The internal-level schema represents how the data are actually stored and accessed.)
   d. logical-level schema (Incorrect. This is not a schema mentioned in the text.)

5. Which of the following would managers most likely use to retrieve information about sales during the month of October?
   a. DML (Incorrect. DML—data manipulation language—is used for data maintenance.)
   b. DSL (Incorrect. DSL is not a DBMS language.)
   c. DDL (Incorrect. DDL—data definition language—is used to build the data dictionary, create a database, describe logical views, and specify any limitations or constraints on security.)
   ▶ d. DQL (Correct. DQL—data query language—is used to retrieve information from a database.)

6. Which of the following attributes would most likely be a primary key?
   a. supplier name (Incorrect. The primary key must be unique. The same name could be used by multiple entities.)
   ▶ **b.** supplier number (Correct. A unique number can be assigned as a primary key for each entity.)
   c. supplier Zip code (Incorrect. The primary key must be unique. More than one supplier could reside in the same Zip code.)
   d. supplier account balance (Incorrect. The primary key must be unique. The same account balance, such as a $0.00 balance, could be maintained by multiple entities.)

7. Which of the following is a software program that runs a database system?
   a. DQL (Incorrect. DQL—data query language—is used to retrieve information from a database.)
   ▶ **b.** DBMS (Correct. A DBMS—database management system—is a software program that acts as an interface between a database and various application programs.)
   c. DML (Incorrect. DML—data manipulation language—is used for data maintenance.)
   d. DDL (Incorrect. DDL—data definition language—is used to build the data dictionary, create a database, describe logical views, and specify any limitations or constraints on security.)

8. The constraint that all primary keys must have non-null data values is referred to as which of the following?
   a. referential integrity rule (Incorrect. The referential integrity rule stipulates that foreign keys must have values that correspond to the value of a primary key in another table or be empty.)
   ▶ **b.** entity integrity rule (Correct. Every primary key in a relational table must have a non-null value.)
   c. normalization rule (Incorrect. The text does not discuss a normalization rule.)
   d. relational data model rule (Incorrect. The text does not discuss a relational data model rule.)

9. The constraint that all foreign keys must have either null values or the value of a primary key in another table is referred to as which of the following?
   ▶ **a.** referential integrity rule (Correct. The referential integrity rule stipulates that foreign keys must have values that correspond to the value of a primary key in another table or be empty.)
   b. entity integrity rule (Incorrect. This rule states that every primary key in a relational table must have a non-null value.)
   c. foreign key value rule (Incorrect. The text does not discuss a foreign key value rule.)
   d. null value rule (Incorrect. The text does not discuss a null value rule.)

10. Which of the following attributes in the Cash Receipts table (representing payments received from customers) would most likely be a foreign key?
   a. cash receipt number (Incorrect. A cash receipt number is a good candidate for the primary key of the Cash Receipts table.)
   b. customer check number (Incorrect. Because there is no reason to store customer check numbers in a separate table, it is not a good candidate for a foreign key.)
   ▶ **c.** customer number (Correct. Customer number would be a foreign key in the Cash Receipts table and would link the Cash Receipts table to the Customer Table.)
   d. cash receipt date (Incorrect. Dates usually are not good candidates for foreign keys. The cash receipt date would likely be an attribute in the Cash Receipts table.)

Comprehensive Problem Solution

Since lender and appraiser data are repeated throughout Table 4-14, the spreadsheet contains update, insert, and delete anomalies. To eliminate anomaly problems and reduce redundancy, we break the spreadsheet into three smaller tables: borrowers (Table 4-20), lenders (Table 4-21), and appraisers (Table 4-22).

TABLE 4-20 Borrower Table

Borrower Number (Primary Key)	Last Name	First Name	Current Address	Requested Mortgage Amount	Lender Number (Foreign Key to Lender Table)	Property Appraiser Number (Foreign Key to Appraiser Table)
450	Adams	Jennifer	450 Peachtree Rd.	$245,000	13	8
451	Adamson	David	500 Loop Highway	$124,688	13	9
452	Bronson	Paul	312 Mountain View Dr.	$345,000	14	10
453	Brown	Marietta	310 Loop Highway	$ 57,090	15	10
454	Charles	Kenneth	3 Commons Blvd.	$ 34,000	16	8
455	Coulter	Tracey	1367 Peachtree Rd.	$216,505	13	8
456	Foster	Harold	678 Loop Highway	$117,090	12	9
457	Frank	Vernon	210 Bicayne Blvd.	$ 89,000	12	10
458	Holmes	Heather	1121 Bicayne Blvd.	$459,010	16	10
459	Johanson	Sandy	817 Mountain View Dr.	$ 67,900	15	9
460	Johnson	James	985 Loop Highway	$ 12,000	12	10
461	Jones	Holly	1650 Washington Blvd.	$ 67,890	15	9

TABLE 4-21 Lender Table

Lender Number (Primary Key)	Lender Name	Lender Office Address
12	National Mortgage	750 16 St.
13	Excel Mortgage	6890 Sheridan Dr.
14	CCY	28 Buckhead Way
15	Advantage Lenders	3345 Lake Shore Dr.
16	Capital Savings	8890 Coral Blvd.

TABLE 4-22 Appraiser Table

Property Appraiser Number (Primary Key)	Property Appraiser Name
8	Advent Appraisers
9	Independent Appraisal Service
10	Jones Property Appraisers

Borrower number, lender number, and appraiser number are the primary keys because each uniquely identifies the rows in their respective tables. The primary keys from the Lender and Appraiser tables are added to the Borrower table as foreign keys so that the Lender and Appraiser tables will have a direct link to the Borrower table.

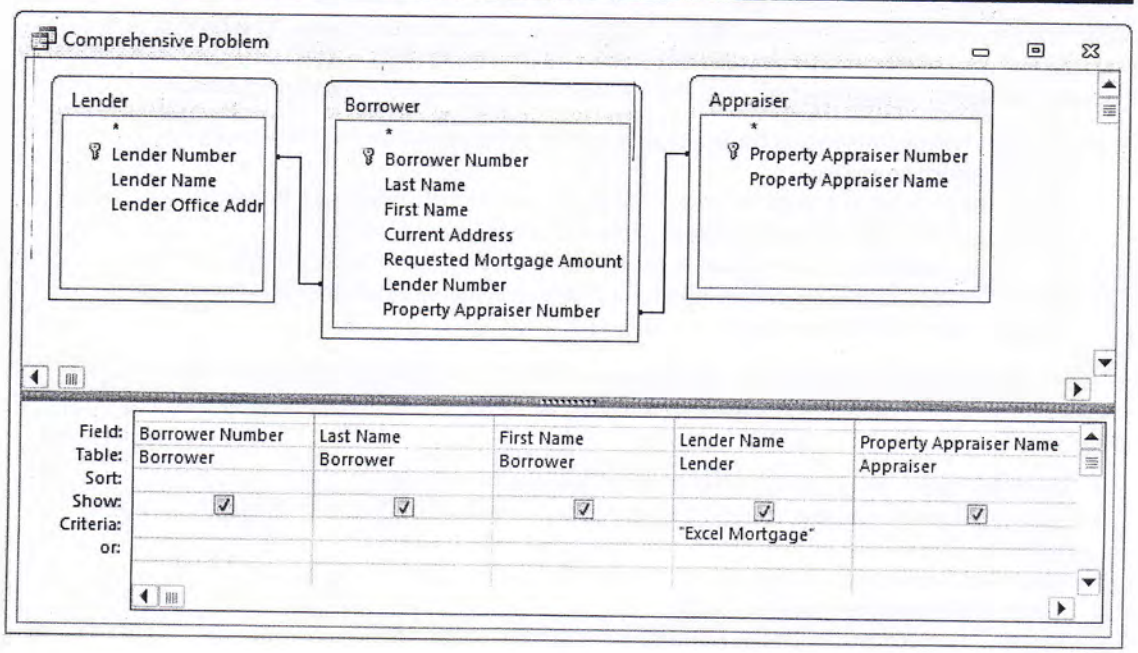
Creating smaller tables with primary and foreign keys solves the three anomaly problems.

- The insert anomaly is solved because a new lender and appraiser can be added without requiring a borrower.
- The delete anomaly is solved because deleting a borrower that decides not to pursue a mortgage does not delete information about the lender and appraiser.
- The update anomaly is solved because there is only one row in one table to update when a lender moves and changes its address, instead of changing all spreadsheet rows that store the lender address.

After the data are entered into the Microsoft Access tables, we can query the database. The query in Table 4-23, which finds the borrowers and appraisers associated with loans from Excel Mortgage, is created as follows:

- From the Query menu option, select "Create Query in Design view."
- Add all three tables to your Query Design. Access automatically links the primary and foreign keys.
- Select the following fields: Borrower Number, Last Name, First Name, Lender Name, and Property Appraiser Name.
- Specify "Excel Mortgage" as the criteria in the Lender Name column.
- Run the query.

TABLE 4-23 Borrowers with Loans from Excel Mortgage



Query Answer