Chapter 1

Introduction to Microsoft Access

Notes to the Instructor

This chapter introduces the Microsoft Access 2010 database management software. Students will learn about using tables, displaying database information, finding answers to questions with database queries, using forms, and printing database reports. Some students already have an adequate working knowledge of Microsoft Access and therefore this chapter may be unnecessary review for them. We suggest that you poll your students to determine their experience levels before proceeding. If you determine that the class has sufficient experience, it may be best to proceed to Chapter 2, Databases and Accounting Systems.

The databases for this book are available online from Cengage Learning at www.cengage.com/accounting/perry. The site contains complete files that aid students in their learning process. We refer to these files in the process of completing the textbook exercises.

The many exercises presented in Chapter 1 convey the chapter's content via a hands-on approach. It is important that the students read and carefully follow each exercise. The exercises sometimes build on one another. The exercises are an important part of the carefully crafted student learning experience.

Lecture Outline

The pages that follow provide an outline of the contents of Chapter 1.

Objectives

- Understanding the Access work environment.
- Creating and using Access objects including tables, queries, forms, and reports.
- Customizing the Access environment.
- Opening and displaying tables.
- Retrieving information with queries.
- Modifying tables' contents with action queries.
- Creating and using forms to display data.
- Designing and using database reports.

Introduction

What is Microsoft Access?

- A relational database management system that supports small to medium-sized database applications. Microsoft Corporation produces the database system.

What is a Relational Database?

- Dr. E.F. Codd, working for IBM at the time, is the acknowledged father of modern relational database management systems.
- A relational database system normally contains tables, queries, forms, and reports.
- A table is a two dimensional object having rows and columns and resembling a worksheet.
- Each column of a table lists a different characteristic known as an *attribute.*
- A collection of tables that are related to one another form a database.
- One table may represent employees' personal information, another may represent employees' skills, and another may represent employees' work history.
- When a database consists of a single file, it is known as a *flat file*. (In reality, this situation *rarely* occurs.)

Starting Microsoft Access

- Access 2010 opens in *Backstage View*.
- Backstage View is where you do thing to your databases, whereas the *Access client* is where you deal with a particular database.

Obtaining Help

Exercise 1.1: Displaying Help. This exercise shows students how to locate help. You might want to ensure that they understand how to seek context-specific help so that they can be self-sufficient when expert human advice is unavailable. Search for a couple of different terms to illustrate how help works. Also illustrate that the Access toolbar has *ToolTips*, which are displayed when the mouse hovers over any of the toolbar icons.

Printing Help

- Locate the help screen you would like to print.
- Choose the desired help topic.

Exiting Access

- Exiting Access updates all changes to the database, closes the database, and returns to the desktop or another suspended program.
- Students have an especially difficult time understanding that Access is continually updating database objects and that it is not normally necessary to "save" data before exiting Access. Reassure the students that if an altered object has not yet been saved, Access will display a dialog box asking them if you want to save the changed object.
- Choose Exit from the File tab to leave Access.

Examining the Access Environment

Exercise 1.2: Opening the Chapter 1 Database. This exercise shows the students how locate and open an existing database—the one associated with Chapter 1 Please refer to the textbook exercise.

Access Work Surface

- The Access Ribbon
  - Known as Office fluent interface.
  - Contains command tabs with groups of commands by usage.
  - Home command tab contains commands to format tables (text, font, etc.), create, save, sort, and filter records, and search and replace.
- The Navigation Pane displays a database's objects in categories.
  - Contains categories (Tables, Queries, Forms, Reports, Macros, and Modules) and groups within the categories.
  - Click the open/close button on the Navigation Pane to open or hide it.
  - Custom categories are available in addition to the Object Type and Table and Related Views groupings.
- Setting Access Options allows you to customize the look and feel of access and is available through the Access Options button on the Office Button.

Exercise 1.3: Setting Access Options. This exercise ensures that all students' Access programs have the same global settings such as a tabbed document window, nonoverlapping windows, compact on close, and remove personal information.

- The Quick Access Toolbar provides a fully customizable list of icons that you set for easy access to often needed command buttons.

Access Objects

- Objects are the structures you create and methods you employ to maintain and display your data.
- Objects described in this chapter include tables, queries, forms, and reports.
- Tables (see Figures 1.3 and 1.4, for example):
  - Hold all data in the database. (Stress to the students that only *tables* hold the data—no other of the Access objects, including queries, forms, or reports, actually holds data.)
  - Are two dimensional and have rows and columns.
  - Row ordering is unimportant, as the rows can be sorted and rearranged without changing the fundamental table information.
  - Column ordering is unimportant, meaning that the particular ordering of the columns bears no significance, and *any* table column may be placed in any particular position.
- Queries (Figure 1.8 shows an example of a query that selects total sales in dollars tallied by product description:
  - Access supports several types of queries classified as either *action* queries or *selection* queries. Select queries are the most common type. They pose questions of the database and return answers in a *dynaset.*
  - Subsets of rows are returned when selection criteria are specified to filter the data.

- Forms (Figure 1.9):
  - Provide a way to view table data one row at a time.
  - Facilitate data entry for inexperienced users.
  - Display data from tables or queries but do not actually *hold* data.
  - Have navigation buttons that speed movement from record to record.
- Reports (Figure 1.10):
  - Provide formatted, hard-copy output.
  - Display database information that can be supplied by tables, queries, or both.
  - Can be customized to produce typeset-level output.

Working with Databases and Tables

Opening a Database

Exercise 1.4: Opening the Chapter 1 Database. This exercise shows the students how locate and open the Chapter 1 database. Although Exercise 1.3 did this already, this serves as an introduction to creating a backup of a database. Please refer to the textbook exercise.

Creating a Backup of an Access Database

This is particularly important for students to understand in case they create a database from scratch and don't pay attention to the default folder where Access stores the data. By opening the database from the frequently used list on the opening window, students can then create a backup and thereby save the database on their flash drives or other known location. A Try it exercise reinforces, with steps, creating a backup (copy) of an open database.

Looking at Data through Different Tabs

- Data can be viewed and inspected through a table's Datasheet view, a form, a query's Datasheet view, or a report—individually or simultaneously. In addition, you can view both a form and a datasheet in with a split form (Figure 1.12 shows tabbed document windows of three of these views).
- Emphasize that it is important to learn—memorize at first—these object-naming rules to conform to standard conventions and make dealing with objects consistent and uniform.
- Be sure to review Figure 1.13 because it shows the relationships between the five Chapter 1 tables that comprise the database.

Opening a Table

Emphasize that there are usually several different ways to accomplish a task with Access. Opening a table provides is no different: Double-click a table's name, right-click and select Open, or drag the table's name to the Access work area.

Exercise 1.5: Opening a Table. This exercise shows the students how to open the *tblCustomers* table (Figure 1.14) and examines the Access' status bar icons (Figure 1.15). The table *tblCustomers* contains information about Incredible Cheesecake Company customers.

Take some time to explain the command tabs and the groups they contain. Figures 1.15 and 1.16 have good graphics that you can point to while explaining the fluent interface ribbon and its contents.

Moving Around a Table

- When you display a table, there are several ways to move from one record to the next. You can:
  - Select the Home tab, click the Go To command, and then click *First*, *Previous*, *Next*, *Last*, and *New*.
  - Use the keyboard PgUp, PgDn, or arrow up and down keys.
  - Use the table navigation buttons that appear in the bottom edge of the window.
- A *row pointer* rests in the currently-selected row.
- The leftmost column of the table's datasheet contains a column called the *record selector column* (this column is not stored in the table).

Searching for a Value in a Column

- Use the Find command to locate a specific field value in a large table.
- Exercise 1.6 shows you how.

Exercise 1.6: Searching for a Row Containing a Particular String. This exercise shows the student how to quickly locate a customer whose street address contains a particular string. Please refer to the textbook exercise for details of how to do this.

- Discuss with students what happens if the search value cannot be found so they understand the difference between "successful" and "unsuccessful" searches.

Changing a Table's Display Characteristics

- You can change a table's *display* characteristics by choosing one or more columns and then clicking the right mouse button; the table's structure remains unchanged.
- If you close a table after changing its display characteristics (but not its structure), Access prompts you as to whether or not you want to save the table's display characteristics.
- Remind students that saving altered table display characteristics isn't particularly important because doing so merely saves the way the table is portrayed on screen.

Exercise 1.7: Changing a Table's Display Characteristics. This exercise shows students how to change a column's displayed position and width in the *tblInvoices* table (Figure 1.17). Remind students that changing a table's display characteristics including column positions, widths, etc. is a cosmetic change that does not affect the underlying table's column order or other properties. Please refer to the textbook exercise.

Sorting and Filtering Table Rows

- You can locate a particular record much more easily when the table's records are sorted and when they are filtered to display only a particular subset based on one or more criteria.
- You can use a quick sort to organize a table's rows based on one or more columns; sorting commands are found on the Sort & Filter group of the Home command tab.
- Applying filter allows you to restrict rows that appear based on the criteria you specify for a filter.
- A rich variety of filters are available on a column's drop down list including extensive number and text filters as well as blank and specific-value filters.

Exercise 1.8: Sorting and Filtering Table Rows. This exercise shows students ways to sort the tblInvoices table in ascending order by the Shipper column. The column drop-down list is used to select a unique ship date from the list. The filter restricts the rows to that date. Please refer to the textbook exercise.

Explain the Toggle Filter and Clear All Sorts buttons and their use. The Sort & Filter group of the Home command tab holds those commands.

Printing a Table

- Print a table by selecting the table (open or not) in Datasheet view and then accessing printing commands through the Backstage View. It is accessible by clicking the File tab.
- Choose any desired options in Print dialog box and then click OK.
- You can print detailed information about the definition of a table by executing a slightly more complicated sequence of steps. This is probably not as important to students as simply printing a table's datasheet but is included for completeness.

Printing a Table's Structure

- Print a table's structure is a less-often used activity, but it is necessary at times.
- Click Database Tools and then lead the students through a small example of making selections in the Database Documenter found in the Analyze group.

Querying a Database

- Querying a database is posing a question which returns an answer in the form of a virtual table known by the name Access-specific name of *dynaset*.
- A query is a stored definition that specifies all tables involved in data retrieval, which columns are to be retrieved, which records are to be included in the result, and any calculations to be performed (see Figure 1.19).
- Most queries are called *selection* queries because they retrieve selected rows from tables
- Other queries do not return answers, but insert new records, delete existing records, update data, and create new table columns. These are known as *action queries.*
- Access uses QBE (query by example) to define queries.
- Use the industry-accepted object naming convention to name all queries: start with the prefix *qry* followed by initial capitalized words representing the remainder of the name. For example, *qryFindHighSales* (without spaces).

Using a Query

Exercise 1.9: Running a Query. This exercise shows students how to open and run an existing query, *qryArcataCustomers*. Remind students that they can run an existing query—one that is stored in the Queries category on the Navigation Pane. Right-click the query name and click Open or by double-clicking the query name. (We prefer the latter method because it involves fewer actions.) Please refer to the textbook exercise for further details.

Creating a One-Table Query

- Queries that restrict which rows are returned by using some criteria (e.g., customer invoices over 59 days old) are the basis of a fundamental relational database operation called *selection* (you specify which rows to select).
- You can form one-table queries by selecting a table and then placing column names in the QBE grid corresponding to the columns you wish to see. You can specify selection criteria to restrict the returned rows to those which satisfy the criteria.

Exercise 1.10: Creating a One-Table Query. This exercise shows the students how to search for and select a specific customer from the tblInvoices table and return rows based on a date filter and a shipper filter—all supplied in the criteria row of the QBE grid. (See Figures 1.20 and 1.21.)

Be sure to point out to students that date values must be contained between # symbols. They don't always pay attention to that detail, and Access will not compensate for their omission.

Saving a Query

- Queries can be saved and re-executed periodically to produce current lists of clients, spare parts, invoiced over 60 days past due, etc., reflecting changes in the underlying table(s) over time.
- With more than one query open, only the active query—the one whose tab is active—is saved.
- The dynaset *cannot* be saved directly, but you can turn the query generating the dynaset into a so-called *Make Table* query. Generally, however, you want to save only the query and not its results.
- A Try It example encourages students to save a query they created in Exercise 1.10. (You should encourage students to do all the "Try It" examples. Doing so will sharpen their skills).

Sorting the Results

- The normal, default row order of a dynaset is ascending order by the table's primary key.
- If a table has no primary key, then table rows are displayed in no particular order.
- A dynaset can be sorted by any field or combination of fields by indicating which columns are the sort columns in the query's definition.
- Select and order columns according to sort criteria—the first column having sorting selected is primary sort column while subsequent columns similarly marked are secondary (and beyond) sort columns.
- Select a sort option in the sort row of the QBE grid from the drop-down list.

Using More Complex Selection Criteria

- Queries that must satisfy more than one condition, *with all conditions being true*, use the AND operator (e.g., discount percentages AND how many occurred in January).
- Figure 1.23 shows a list of comparison operators used by Access to compare two strings or numerical values.
- Figure 1.24 displays the list of Access' logical operators.
- Comparison operators are <, >, =, <=, >=, and <>.
- Logical operators are *And*, *Or*, and *Not*.

Exercise 1.11: Writing a Query with an "AND" Operator. This exercise shows the students how to use the AND operator to query and display invoice lines with discount percentages greater than zero and less than 11 (Figure 1.22). Be sure to point out to students that numeric criterion values are not enclosed in quotation marks.

Creating Selection Criteria Using the "OR" Operator

- When any one of several criteria must be satisfied for a row to appear in a dynaset, then you use the OR logical operator. For example, you use OR to return rows from the *tblProducts* table when any product description contains "choc" (as in *chocolate*) or the retail price is less than $10

Exercise 1.12: Forming a Query with "OR" Criteria. This exercise shows students how to form a query involving two different criteria columns of the *tblProducts* table (Figure 1.25). Walk through this example in class. It is a good example of a relatively complex query involving only one table.

- Explain the use of the wildcard character * on each end of choc so that the string "slides" across all description strings for a match anywhere in the string.
- Queries having two or more independent selection criteria column values have as many criteria rows in the query grid as there are individual criteria for which any one being true is sufficient to satisfy the condition.
- Queries with alternate selection criteria on *one* field (e.g., "Chocolate" Or "Vanilla" in the ProductDescription column): write both values (complete strings in this example) in the same Criteria row and separate the two values with *Or*.
- An asterisk in a criterion (e.g., "*choc*") for character string criterion is called a *wild card character*, which allows extensions like *white chocolate* and *chocoholic* both to be included in the dynaset.
- Access ignores capitalization and locates matching rows based on spelling alone. This is a very important point that students often overlook. Highlight this fact. It doesn't matter, for instance, whether you specify the criterion *Chocolate*, *ChoCoLAte*, or *chocolate*. Each returns a row in which the characters *chocolate* occur (spelled correctly, with nothing else in the string), regardless of capitalization.

Including Expressions in a Query

- Expressions are special "fields" that display the results of calculations in a dynaset. Of course, expressions *cannot* be stored in a table, because a table stores only values (invariants), not expressions (unlike a spreadsheet).
- Access allows an expression in a *table* field. We discourage using this because it violates normalization rules and is, frankly, sloppy design.
- Query expressions are based on existing table or query fields.
- Query expressions are also known as *virtual fields*.
- A Try It example shows students a query, *qryInvoices*, which calculates totals and displays a new, virtual column in the dynaset (Figure 1.26).

Printing Dynasets

Printing a dynaset is straightforward:

- Display the dynaset.
- Check the dynaset to ensure it is what you expect.
- Click the File tab, click Print, and click Print.
- Make necessary selections in the Print dialog box and then click OK.

Using Forms

- Forms provide convenient, less cluttered work surfaces through which you can enter or alter information in tables and queries.
- Forms can be created that facilitate data entry and mimic already familiar paper forms.

- Forms facilitate validation checks on entered data.

Viewing a Table through a Form

Exercise 1.13: Opening an Existing Form. This exercise shows students how to open and move around the *frmProducts* form found in the Chapter 1 database **Ch01.accdb** (Figure 1.27).

- Form navigation buttons allow movement through rows of table.
- A form presents an intuitive and pleasant looking interface that is often familiar to a user.

Viewing a Query through a Form

- A Form can be created from a saved query—*qryFebFedEx* in this case (Figure 1.28).
- The total number of records displayed in a form—one after another—depends on how many rows are produced by query.

Creating a Form Quickly

- A Try It exercise shows students how to create a quick form using the Form button in the Create command tab Forms group. It is based on the *tblCustomers* table found in the Chapter 1 database, Ch01.accdb (Figure 1.29).
- You can alter the form after you create it. Students should know that the generated form could be redesigned. The default form serves as a convenient baseline form design.

Saving a Form

- You save a newly created form design, not the data displayed within it, by clicking the Save button on the Quick Access Toolbar. Any table *data* altered through a form is preserved automatically by Access when you close a form, move to another record, or close Access.
- A Try It exercise shows students how to save a form.

Editing Data with a Form

- It is often easier to alter data in a table by using a form view of the table data.
- Usually, a form displays one record at a time.
- In Form view, click the field you want to change, make any changes, and move to another record to post the changes. Alternatively, you cause the changes to be posted when you close the form.

Exercise 1.14: Editing Data with a Form. This exercise shows students how to edit data using the *frmCustomers* form. You can change a field by moving to it and then entering a new value.

- Remind students that changes made to a record are *not* posted to the database until you move to another record. In the case of a form, simply press PgDn (or PgUp) or click Tab a sufficient number of times to move to the next record.

Creating and Using a Split Form

- Split forms display data in two views on a single document window: a Form view on the top half and a Datasheet view on the bottom half.
- Modifying data in either pane of a split form modifies it simultaneously in the other view.
- Split forms provide the advantages of a Datasheet view and Form view in one document.

Exercise 1.15: Creating a Split Form. This exercise shows students how to create a split and altering the size of the panes by moving the form splitter bar. The split form is based on the table *tblCustomers* and is shown in Figure 1.30.

Printing a Form

- Print one record by locating it within the form (use navigation buttons) and clicking the Office Button, pointing to Print, clicking Print, choosing "Selected Record(s)" radio button, and clicking OK in Print dialog box.
- Print a range of form pages by clicking the Pages option button and then entering the beginning and ending page numbers to print.

A frequent problem that arises when students want to print a form is that several forms usually appear on a page. The trick to printing only a single form is to place a page break just below the last field of the form (in design view). However, the students have not yet learned about the toolbox in Design view, so you can merely mention it at this point. Alternatively, selecting the form and choosing the "Selected Record(s)" option is still the best way to print a single form on a page.

Designing Reports

- You cannot enter or edit data in a report displayed in Print Preview.
- You can filter report data when the report appears in Report view.
- Reports are a way to produce hard copy output—to display database information—in an orderly manner.
- Reports can be created ranging from simple utilitarian to professional designs with multiple typefaces, drop shadows, graphics, and colors.

Previewing a Report

Exercise 1.16: Loading and Previewing a Report. This exercise shows students how to load the existing report *rptCustomerInvoices*, examine it in Report view, and close it (Figure 1.31).

Creating a Report Quickly

- Creating a basic report of the data in a table or table is simple: Select the table name in the Navigation Pane, click Create, and click Report to produce the default (and simplest) report.
- Reports can be based one or more tables or a query.

Exercise 1.17: Creating a Report Quickly. This exercise shows students how to create a basic report based on the table *tblCustomers* (Figure 1.32)

Saving a Report

- Right-click the report's document tab and click Save.
- Select and enter the report's name, remembering to precede the name with the standard report prefix, *rpt* (e.g., *rptCustomers*).

Exercise 1.18: Saving a Report. This exercise shows students how to save a newly created report. Right-click the report document window's tab, type the report's name, and click OK. The report's design is saved among the Report objects.

Exercise 1.19: Deleting a Report. This exercise shows students how to delete a saved report from the Navigation Pane. Click the report's name in the Navigation Pane. Press Del (keyboard key), and click Yes in the confirming dialog box.

It may be easier for students to use the right-click menu for *all* Navigation Pane object operations including delete. Right-click the object in the Navigation Pane and click Delete from the pop-up menu. This is probably a better choice—right-clicking—because several alternative actions are proposed in one pop-up menu.

Printing a Report

A Try it exercise leads students through opening a report and navigating to various report pages in Print Preview view. Then, they print one of the several pages. It is important that students are aware that the entire report could be long and that the Print dialog box Pages options should be chosen to restrict the number of pages printed. An exercise leads students through closing their Chapter 1 database.

Exercise 1.20: Closing the Database and Closing Access. This exercise shows students how to close the database and close Access simultaneously

- Click the File tab and then click Exit.

Answers to Review Exercises

Multiple Choice Questions

1.	c	6.	c
2.	b	7.	d
3.	d	8.	c
4.	a	9.	b
5.	c	10.	b

Discussion Questions

The solutions presented here come from the textbook discussion. Your students may include other insight that is relevant, but not presented in our solutions.

1. Discussion should involve at least mention of the Microsoft Fluent Interface—the ribbon, its groups of commands, and contextual tabs. Creating backups via the Microsoft Office Button is important also.

2. Occasionally, it is better to work with table data one row at a time. Tables are not a very intuitive interface for some people, especially those who are not computer literate. Access forms solve this problem. Forms let you see the data from a table in a format that is easier to understand. You can see one row or many rows of a table. Figure 1.9 shows an example of a form displaying a record from the *tblCustomers* table more attractive and intuitive environment. (For more information on forms, please see the discussion about forms.)

3. Two of likely several methods are these:
   - Method One:
     - Right-click a table name in the Navigation Pane.
     - Click Open in the pop-up menu.
   - Method Two:
     - Double-click a table name in the Navigation Pane.
   - Method Three:
     - Drag the table name from the Navigation Pane onto the Access work area.

4. One of the real advantages of relational databases is the ability to ask questions that return interesting and meaningful answers derived from a database. Relational database systems make asking questions particularly easy, and Access is no exception. A *query*, the usual name for a question, can be simple or complex and can involve only one table, dozens of tables, or even hundreds of tables. In a query, you specify which tables are involved in the data retrieval operation, which columns are to be retrieved, which records are to be returned, and any calculations to be performed. Most queries are called *selection queries*, because they retrieve rows from tables.

5. Producing reports is as easy as clicking a button or two. Frequently, you will want to either preview a report on screen or produce a printed report that you can pass around at a meeting or keep as a permanent record. Access reports are just that—reports. You can neither enter nor edit data in a report. You can create reports ranging from simple, utilitarian designs to professional looking reports replete with attractive typefaces, drop shadows, and graphics. Reports typically display information from a table, a collection of related tables, or a query. Though a lot can be gained from looking at the query's results on screen, it is even more useful to have a printed report. (For more information on reports, please review the *Designing Reports* section.)

Practice Exercises

A note to the instructor: We have created, in the Navigation Pane of the instructor's version of the database, a custom category called *Review Questions: Answers*. That custom category contains three custom groups: *Practice Exercises*, *Problems*, and *Student Database*. You will find all the solutions to the practice exercises in the first group and all answers to the problems in the second one. If the custom category *Review Questions: Answers* does not appear in the Navigation Pane, click the category name at the top of the Navigation Pane and click the *Review Questions: Answers* category name.

1. A solution is available on the instructor's solution database in the Practice Exercises custom group. It is called *1-Practice Exercise 1*. (It is a *copy* of the *tblCustomers* table, but in the specified order.) The trick is to move the City column to the left of the CustomerName column (drag City to the left of CustomerName). Next, drag through the column headers of City and CustomerName and click the Ascending button in the Sort & Filter group of the Home command tab.
2. A solution is available on the instructor's solution database in the Practice Exercises custom group. It is called *1-Practice Exercise 2*.
3. A solution is available on the instructor's solution database in the Practice Exercises custom group. It is called *1-Practice Exercise 3*. Unlike students' solutions, we have set the table property *Filter On Load* property to True so that the filter is applied automatically when you open the table in Datasheet view. Students create the filter by clicking the down-pointing arrow on the Retail Price column, point to Number Filters, click Greater Than, and type 25 in the text box of the Custom Filter dialog box. Sort the resulting rows by right-clicking the ProductDescription column and then clicking Sort A to Z.
4. A solution is available on the instructor's solution database in the Practice Exercises custom group. It is called *1-Practice Exercise 4*. We have set the print orientation to *Landscape* and then saved the object. You will have to navigate to record 3 manually and print only that record. The problem as stated in the textbook pretty well lays out the steps to do this.
5. A solution is available on the instructor's solution database in the Practice Exercises custom group. It is called *1-Practice Exercise 5*. Open the Navigation Pane of your copy of the database to locate it. You will have to restrict the report to one page yourself by selecting the Pages radio button in the Print dialog box and then typing "1" (without the quotation marks) in the From text box.

Problems

1. The answers to the three part question are stored in your database in the Problems custom group. They are called *1-Problem 1A*, *1-Problem 1B*, and *1-Problem 1C*. Open up these queries in Design view to review their construction.
2. The answer to the question is stored in your database in the Problems custom group. It is called *1-Problem 2*. Open the form in Form view, click the Record Number text box, type 4321, and press Enter to move to that record. Printing: Click the Office Button, point to Print, click Print, and click the Selected Record(s) button. Click OK to print just the selected record.
3. The answer to the question is stored in your database in the Problems custom group. It is called *1-Problem 3*. Double-click the report to open it in Report view. Printing: Click the Office Button, point to Print, click Print, click the *From* text box, and type *2*. Click OK to print page 2 of the report.

Chapter 2

Databases and Accounting Systems

Notes to the Instructor

In this chapter, students learn about database accounting systems and how they differ from double-entry bookkeeping systems. It explains the connections between accounting systems and database systems, why a relational database system is superior to double-entry bookkeeping systems for capturing detailed accounting information. Upon finishing this chapter, each student will understand how to create efficient, optimal database objects to capture accounting information Completed instructor files are available for use in a protected area of the companion Web site.

Before beginning, download the Chapter 2 database, **Ch02.accdb**, from this book's companion Web site. (We assume you can do this part on your own computer.) Your instructor's version of the Chapter 2 database contains answers, where appropriate, for the end of chapter questions. You may *not* want to distribute to students.

Lecture Outline

This chapter is independent of other chapters in the book in that it provides a theoretical foundation that links accounting systems and database systems.

Objectives

- Chapter 2 presents the theoretical foundation for the remainder of the text and contains both practice and theory for both accounting transaction cycles and database management systems
- Important topics covered include:
  - Differences between double-entry bookkeeping and database accounting systems.
  - Advantages and disadvantages of database accounting systems.
  - Business processes.
  - The relationship between accounting systems and database systems.
  - A brief history leading to the development of database management systems.
  - Functions of database managements systems.
  - Theory and application of relational database management systems.
  - The structure of database objects that store accounting events.
  - The importance of normalizing tables.
  - Performing database selections, projections, and joins.
  - The resources-events-agents (REA) model for accounting databases.
- The Coffee Merchant, a fictitious coffee bean and tea wholesaler, demonstrates the classic accounting application involving processing and maintaining invoice data
  - Keeping track of unpaid customer invoices
  - Generating relevant reports based on current and historical customer data

Introduction

- Students usually learn accounting using manual systems with journals and ledgers.
- This chapter explains the differences between database accounting systems and double-entry bookkeeping.
- Database accounting provides advantages, yet also has certain disadvantages.

- The business activity classifications in the chapter provide a three-level taxonomy of complexity that can help system designers decide when to incorporate certain features in their designs.
- Transaction cycles provide accountants and auditors a way to organize economic events into related categories.

Database Accounting Systems

- Chapter 2 introduces database theory and explains how firms can use relational databases as part of their accounting systems.
- Businesses see the advantages of relational databases for all their information processing needs.

Events-Based Theories of Accounting

- Researchers such as William McCarthy and George Sorter developed events approaches to accounting that lend themselves to the use of relational databases to perform accounting tasks.
- Events approaches to accounting argue that accountants ideally should store all relevant attributes of economic events in a readily accessible form.
- This is not a fully attainable objective with currently available technology; however, relational databases do a fairly good job of storing many information attributes and giving users a variety of ways to retrieve those attributes.

Double-Entry Bookkeeping Versus Database Accounting

- Double-entry bookkeeping satisfied accountants' need to capture transaction essentials for many years.
- Five hundred years ago, the costs of gathering and storing information were very high.
- Double-entry bookkeeping let businesspersons capture key attributes of transactions in a highly aggregated form, which minimized the cost of information gathering and storage.
- Debit-credit balancing checks provided important internal control in manual accounting systems.
- A Try it exercise asks students to make a trip to an office supply store in search of two-column accounting paper and predicts that the sales clerk will look up the product's location using the store's inventory database, which will most likely be a relational database.
- Computers now easily capture a wide variety of information about transactions with bar code scanners and other devices.
- A sales transaction provides the basis for comparing double-entry bookkeeping and database accounting. The double-entry bookkeeping journal entry stores five attributes of the transaction.
- Figure 2.1 shows how the database accounting system can efficiently store many more attributes of the sales transaction than can be stored in a traditional double-entry bookkeeping system.
- The chapter refers to the normalization rules discussed in Chapter 2 and presents a Try it exercise that illustrates the problems presented by shipping firm names stored in a non-normalized table. This is a review point and is usually worth some class discussion time.
- We note that the database system directly records only some of the items that the double-entry bookkeeping system records and describe how the database system would use queries

to calculate some information attributes that the double-entry bookkeeping system stores in an aggregate form.

- A Try it exercise challenges students to identify multiple ways that a user might enter the names of common shipping companies (for example, Federal Express might be entered as *FedEx*, *FEDEX*, *Fed Express*, *FEx*, or *Federal Express*).
- A Try it exercise asks students to identify interesting facts that can be found in or calculated from the database diagrammed in Figure 2.1.
- The database approach can do everything that double-entry bookkeeping can do and more.

Advantages of Database Accounting Systems

- Reduce data storage costs.
- Eliminate data redundancy.
- Eliminate data inconsistencies.
- Avoid duplicate processing.
- Ease add, delete, and update data maintenance tasks.
- Make data independent of applications.
- Centralize data management.
- Centralize data security.
- Database accounting systems offer much greater flexibility in extracting data than flat file double-entry accounting systems. This flexibility leads to other advantages such as:
  - Ease report modifications and updates.
  - Provide ad hoc query capabilities.
  - Facilitate cross-functional data analysis.
  - Permit multiple users simultaneous data access.
- Provide data entry and integrity controls as part of the database management system.

Disadvantages of Database Accounting Systems

- Greater hardware requirements can be more expensive.
- The cost of the database software itself.
- Cost of employing a database administrator (DBA).
- Centralizing management and security control functions creates several drawbacks:
  - System operation becomes critical.
  - Incorrect data entered corrupts many users' work.
  - Territorial disputes over data ownership may arise.
- Accountants tend to distrust any single-entry accounting system, such as a database accounting system, because double-entry bookkeeping is so pervasive in accounting education and practice. Most accountants' first response is to question and fear anything else.

Business Processes

- A value chain is an accounting system viewed as a collection of processes rather than financial statements.
- Transaction cycles are common transactions associated with business processes.
- Business processes include Financing, HR, Conversion, Sales/Collection, and Acquisition.

Sales/Collection Process

- Includes all sales and cash collection activities.
- Transactions recorded include customer orders, sales, and cash receipts.

Acquisition/Payment Process

- In a manufacturing firm, the acquisition and payment processes includes all activities related to ordering materials, receiving materials, and paying for them.
- Service firms record purchases of materials that aid service including office supplies.

Human Resources Process

- HR processes revolve around employees and include calculating employee pay, deductions, and net pay.
- HR reports include checks to employees and vendors, commission reports, timecard reports, and payroll registers.

Financing Process

- Cash receipts and payments related to equity and debt financing fall are part of the financing process.
- Recorded financing transactions involve issuance of stock, receipt of cash for stock, declaration of dividends, and payment of dividends.

Accounting Information Systems and Database Systems

- Historically, accounting information has been captured in journals and posted to ledgers.
  - Strict rules were applied to the process of gathering data and presenting information.
  - Standard reports (e.g., Income Statement or Balance Sheet) provided a specific set of information with a format and level of aggregation that was determined before the reports were generated.
  - Off-the-shelf accounting software provided these standard reports.
- Modern accounting systems are usually built on relational databases and allow management to generate customized reports that present relevant information for decision making.
  - More information about an event (e.g., a sale can include dollar amounts, customer information, inventory information, time of the sale, and so on) can be captured and easily accessed.
  - Advances in inexpensive and easily learned relational database management software and computer hardware support this approach.
- Events account records details about economic events in a database including who was involved, when the event occurred, and what resources were affected.

Database Management Systems

- Database management systems (DBMSs) are valuable to business enterprises because they provide the software to store, retrieve and modify crucial business data.

Pre-DBMS Data Acquisition and Reporting

- Corporate information (e.g., accounts receivable) was kept by department in flat files on computer disks.
- Standard reports (e.g., inventory stock levels) were readily available.

- Custom or unusual reports had to be specially designed and written, often a time consuming and expensive task.
- Duplicate information, data files mirroring master file information, was often kept by different departments and led to data redundancy and inconsistency (e.g., customer address information was kept by marketing and accounts receivable).

Functions of a Database Management System

The capabilities that a database management system provides in development of an information system are:

- Efficient data storage, update, and retrieval.
- User accessible catalog.
- Concurrency control.
- Transaction control.
- Recovery services.
- Security and authorization services.
- Integrity facilities.

Advantages of Database Management Systems

- Subschema: Provide each user with particular or unique view (authorized access) of the database.
- Data independence: system's ability to hide the details of the physical storage of information from the application programs, or the user's views of access techniques and methods.
- Structure changes are transparent.
- Data sharing: centrally stored information is current and consistent and everyone can have access.

Disadvantages of Database Management Systems

- Secondary (disk) storage—DBMS's require more storage space than flat files.
- Additional people—database experts may be needed to operate system.

Relational Database Management Systems

- Implementation of RDBMS requires a data model—an abstract representation of a database system providing a description of the data and methods for accessing the data managed by the database.
- Three models have been used during the history of databases:
  - Hierarchical—(1960s through early 1970s).
  - Network—(1960s through early 1970s).
  - Relational—(1970s) overwhelming choice today and has many advantages:
    - Logical and physical characteristics are distinct and provide the user a more intuitive view of data.
    - Requires little training.
    - More powerful retrieval and update operators available.
    - Powerful tools to warn when a database has design flaws.
- Relational or RDBMS model is easily the most popular of the three models, and it will be discussed from this point on

Database Objects

- RDBMS model defines the conceptual view that the user has of all the objects contained by the database system.
  - The RDBMS model represents both data and relationships between them.
  - All data, including the database table definitions and object information, exists in tables.
- Relational Database is a collection of relations.
  - A table is a relation that consists of rows (tuples) and columns (attributes).
  - Files, records, and fields are the same as relations, tuples, and attributes (see Figure 2.3).
- Properties of relations:
  - The entries in each column of any row are single valued.
  - Each attribute of a given relation has a distinct name, called the attribute name.
  - Every value in a column contains values for that column only, and the values are of like data type.
  - The order of the rows is unimportant.
  - The order (position) of the columns in relation to each other is unimportant.
  - Each row is unique from all other rows in the relation.
- Customer table—Coffee merchant (Figure 2.4)
  - Rows represent customers—each row must be unique
  - Columns represent attributes of each customer, schema for table includes: CustomerID (primary key), CompanyName, PhoneNumber, and Contact
  - Attribute values are within columns

Primary and Foreign Key Attributes

- Primary key is a column (or group of columns) that uniquely identifies a given row—the CustID column, for example (Figure 2.5).
- Data Dictionary is a collection of tables containing the definition, characteristics, structure, and description of all data maintained by the RDBMS.
- Foreign key is an attribute in one table that must match the primary key in another table—the CustomerID column of the tblInvoice table, for example, shown in Figure 2.5.
- tblInvoice and tblCustomer tables can be joined on the CustID/CustomerID fields.
- Each column of a table is an attribute.
- Columns, including the primary key column and any foreign key columns, can appear in any order.
- Microsoft Access does not require that each table have primary key, but each table *should* have one.

Schema of a Relation

- A schema is a representation of a table that lists all its attributes and identifies the primary key and, optionally, foreign keys
- Schema for the Customer table, tblCustomer, with a reduced number of attributes is:

    Customer(<u>CustID</u>, CustAddress, CustCity, CustState)

  where CustID is the primary key and other attributes are listed and separated by commas

Data Dictionary

- Every relational database system has a data dictionary.

- Data Dictionary is a collection of tables containing the definition, characteristics, structure, and description of all data maintained by the RDBMS.
- Information about tables and their attributes is self contained in the database.
- When a user changes a structure element of a table or other database object, the RDBMS automatically changes the object's definition in the data dictionary to keep it constantly current.
- The data dictionary contains the names of all tables, queries, forms, reports, and other objects as individual rows.
- Microsoft Access does not include a built-in facility for creating and managing a data dictionary. Many other RDBMSs that are used in large database installations (such as Oracle) do include a data dictionary. In Access, the user must create a data dictionary, usually in a separate Access or Microsoft Word file.

The Coffee Merchant Tables

- Figure 2.6 shows the schema of all the tables that make up the Coffee Merchant invoicing system.
- The students should study the primary-key-to-foreign-key relationships in Figures 2.7 through 2.9.

Normalization

- Normalization is the process of determining the correct location for each attribute to meet the relational database rules.
- Unnormalized databases lead to redundant, inconsistent, and anomalous information being stored in tables.
- Properly normalized tables necessary to create an invoice—Coffee Merchant
  - **tblCustomer**—customer information; primary key CustID.
  - **tblInvoice**—unpaid invoices; primary key InvoiceID.
  - **tblInvoiceLine**—quantity information; InvoiceID plus InventoryID combine to form composite primary key.
  - **tblInventory**—catalogs all items available; primary key InventoryID.
- Note: Extended price, subtotal, etc., are calculated when the invoice is printed.
- Other secondary tables involved in the invoicing subsystem include these:
  - **tblEmployee**—contains sales employee information; primary key (pk) is EmpID.
  - **tblCountry**—contains country names for the World and coffee/tea export data; pk is CountryID.

First Normal Form

- First Normal Form (1NF) requires that repeating groups (unnormalized table Figure 2.10) be eliminated
- To attain 1NF status, a new table must be created with a composite primary key—for example, Invoice Line [InvoiceID, ItemID, Quantity].
- Least restrictive, this table can be placed into a RDBMS, but is not as good as higher order forms.
- Tables in 1NF (Figure 2.11) pose several problems:

  - New customer cannot be added without an invoice being generated—known as an insertion anomaly—invoice attributes (e.g., Invoice) cannot be null according to relational database rules.
  - Invoice cannot be deleted without deleting customer information—known as a deletion anomaly.
  - Customer table contains a great deal of redundant information (e.g., Company is repeated for each new invoice).
- Problems can be eliminated by altering the table structure and changing it into Second Normal Form.

Second Normal Form

- Second Normal Form (2NF) requires that each non-key attribute be fully dependent on the entire primary key, not just part of it (e.g., Figure 2.11—Total is functionally dependent on Invoice because a value of Invoice determines a single value of Total, not the primary key, CustID).
- Figure 2.16 shows functional dependencies in the Customer table.
- To attain 2NF status, break Customer table into two tables (Customer and Invoice) and add an additional attribute—a foreign key—linking both tables on their keys CustID and CustomerID, respectively.

Third Normal Form

- Third Normal Form (3NF) requires that a table be in 2NF with all transitive dependencies eliminated (that is, all attributes functionally dependent on the single-attribute primary key).
  - Transitive Dependency occurs when, for example, attribute B determines attribute C, and C determines D (Figure 2.14).
  - To attain 3NF status, create a new table containing at least the determinant attribute (For example, Customer [CustID, CustName]).
- 3NF enforces an informal rule stating that a table should store one fact and one fact only.

Table Relationships

- Process of normalizing creates additional tables with relationships maintained by foreign key to primary key links.
- Three fundamental relationships exist between related tables:
  - One-to-one (1–1)—e.g., a master table contains customer information—a related table contains occasional notes about a few customers.
  - One-to-many (1–M)—e.g., a master table contains customer information—a related table has unpaid invoices for each customer.
  - Many-to-many (M–M)—e.g., a master table contains invoice information—a related table contains inventory information. A relationship table is created to represent the M–M relationship in a convenient way that avoids anomalies. This table makes the connection between two M-M tables so that two 1-M relationships are created among the three tables.
- General Rules Governing Relationships Among Tables:
  - Primary keys must not be null.
  - Create a foreign key from the primary key on the one side of the 1–M relationship.

- Many-to-many relationships are handled by creating an additional table—the relationship table—that consists entirely of the parent tables' primary keys. (The relationship table can contain other columns as well.)
- Most one-to-one relationships indicate unnecessary tables in the database design. Normally, you should merge the two tables. Exceptions occur when there are just too many columns in one table, or one group of fields is used far more frequently than another group. Then, you can consider separating them into different tables for efficiency's sake.

Fundamental Relational Database Operations

Select

- Select operator chooses a set of rows from a table (Figure 2.15)—the query selects rows in which the hire date later than 1/1/2008 and delivers answers in table form.

Project

- Project operator returns a subset of columns from one or more tables (Figure 2.16)—the projection indicates which columns, EmpLName and EmpGender, are retrieved in the answer.

Join

- Join provides the ability to pull together data from disparate but associated tables into a single, virtual table based on a common attribute (Figure 2.17)—EmpDivisionID is foreign key in tblEmployee table and primary key in tblEmployeeDivision table. *Instructors note the following*: The tblEmployeeDivision table has been omitted from this edition of the textbook for simplification reasons.
- Equijoin—rows from the two tables are linked or chained together on matching join column values and the join column appears only once in the result
- Outer join—combines rows from two or more tables on the join column, but rows that do not match on the join column are included in the result

Introduction to Database Design

- An important aspect of database design is the care in choosing the tuples and attributes that each table comprises. This activity is called modeling

Developing Entity-Relationship Models

- Another way to model database objects, called Entity-Relationship (E-R) modeling, to represent a business is a graphical approach using three terms to describe a company's information.
  - Entities (nouns) are the objects found in the company (e.g., invoices, purchase orders, etc.)
  - Relationships are the way in which the distinct entities interact or are related to one another.
  - Attributes (adjectives) describe the entities and relationships.
- Entities and relationships are represented by diagrams (Figure 2.18).
  - Rectangles represent entities.
  - Diamonds represent relationships.

  - Lines represent the connections between the two.
  - Digit or letter above line indicates degree of the relationship: 1–1, 1–M, M–M.
- Completed E-R diagrams can be combined into a system E-R diagram—a process called view integration.
  - Process started by placing the most often used entity in the center of the diagram.
  - Lines connect the related entities.
  - Identify and create tables from entities.
  - Relationships between entities are maintained by foreign/primary key linkages between tables.
  - Third table, a relationship table, is created for M–M relationship—it contains the primary keys of both tables.
  - Last step is to normalize the individual tables.

Resources, Events, Agents (REA) Modeling

- In the late 1970s, Professor William McCarthy drew upon the principles of relational database theory developed by E. F. Codd and the entity-relationship modeling principles of Peter Chen to
  - create a modeling approach specifically designed for accounting systems.
  - develop an entity classification system that would allow accountants to use relational databases handle accounting information.
- REA modeling provides categories of entities that accountants can use to classify the entities that appear in accounting systems.
  - Resources are assets such as cash, inventory, and fixed assets.
  - Events are transactions or other occurrences that have accounting effects and include things such as purchases and sales.
  - Events can also be more subtle occurrences, such as the passage of time that causes interest to accrue on a loan or depreciation of a fixed asset to occur.
  - Agents are humans or organizations that interact with resources and events, such as customers, suppliers, and employees
- A number of researchers have proposed extensions to McCarthy's basic model
  - Eric Denna devised a REAL model, which added an entity classification for location
  - Most accounting researchers agree that the REA model has considerable value as an approach to the design of accounting information systems
  - Figure 2.19 shows a list of entities for a purchase cycle accounting system. The entities are classified using the REA model.

Answers to Review Exercises

Multiple Choice Questions

1.	b	6.	c
2.	d	7.	a
3.	b	8.	b
4.	a	9.	d
5.	c	10.	b

Discussion Questions

The solutions presented here come from the textbook discussion. Your students may include other insight that is relevant but not presented in these solutions.

1. Double-entry bookkeeping provided an excellent way of recording transactions for many years. It helped accountants capture the essence of each transaction in a safe, effective manner. The double-entry nature of the approach also provided an automatic internal control feature in manual accounting systems. Since recording transactions with pen and paper was a time-consuming task, double-entry bookkeeping gave accountants a valuable tool that quickly identified essential elements of transactions. Today's computerized transaction processing systems have freed accountants from the need to capture only essential transaction elements and store them in a highly aggregated form. We can capture a wide variety of information about each transaction quite easily using bar code scanners and other electronic data entry devices. The costs of information storage have also dropped dramatically.

2. In manual systems, the dual nature of the accounting debit and credit model provides a built-in error correction mechanism. In automated systems, this same duality is inefficient and serves no control purpose. Manual double-entry bookkeeping systems can be very efficient; however, most computer implementations of double-entry bookkeeping use a flat file processing design instead of a relational database model. Database accounting systems only store data once and avoid the redundancies of flat-file systems. Database accounting systems use normalized tables to avoid anomalies. Anomalies can become very troublesome as the accounting system grows to a size that requires computers to handle transaction-processing volume.

3. Problems of storing data in two or more places include the creation of outdated and redundant data. Years ago, it was common for departments and individuals to create and maintain their own computer files that duplicated information kept in the master files. Duplicate files were maintained in order to access and examine the data with their own programs quickly, rather than incurring the long delays from the already overworked data processing department. Duplicate data files—files mirroring the master file data—leads to inevitable data redundancy and, soon, data inconsistency. For example, Marketing kept its own files, sorted by zip code, of its larger customers so that they could send them advertising pieces. The marketing department hired a bright young programmer to keep current the "best customer" file, and she would write programs to produce mailing labels from the customer file. Problems occurred when the independently maintained customer list fell out of date.

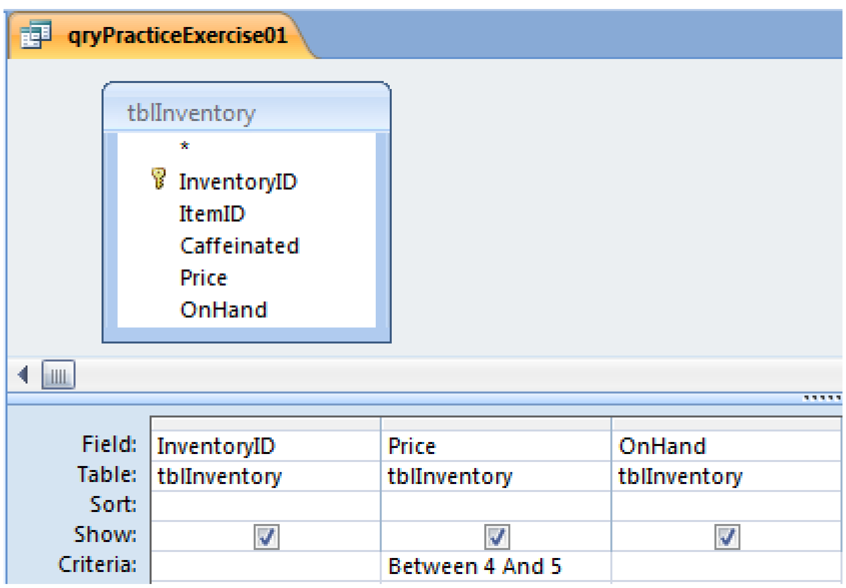
While the master list of customers was kept current by the data processing department from each month's purchase orders, the marketing department did not have access to that data. Their customer list became so out of date—customers moved, and new customers came on board—that it was practically useless.

4. The primary key consists of one or more fields in each table that provides a unique identifier for each row in the table. Every table in a relational database (a table in a relational database is also called a relation) must have a primary key that is unique and that exists for each row in the table. The primary key can include one or more fields (columns). When the primary key includes more than one column it is called a composite primary key. The individual column values in a composite primary key need not be unique, but the combined column values must be unique. The primary key is important because it allows all of the information that has been broken apart to be stored in the individual tables to be pulled back together when a user needs the information. The primary keys (and the foreign keys) provide the connections among all of the tables in a database.

Practice Exercises

The solutions are included in the Ch02-PracticeExercises.accdb file on the Instructor web site.

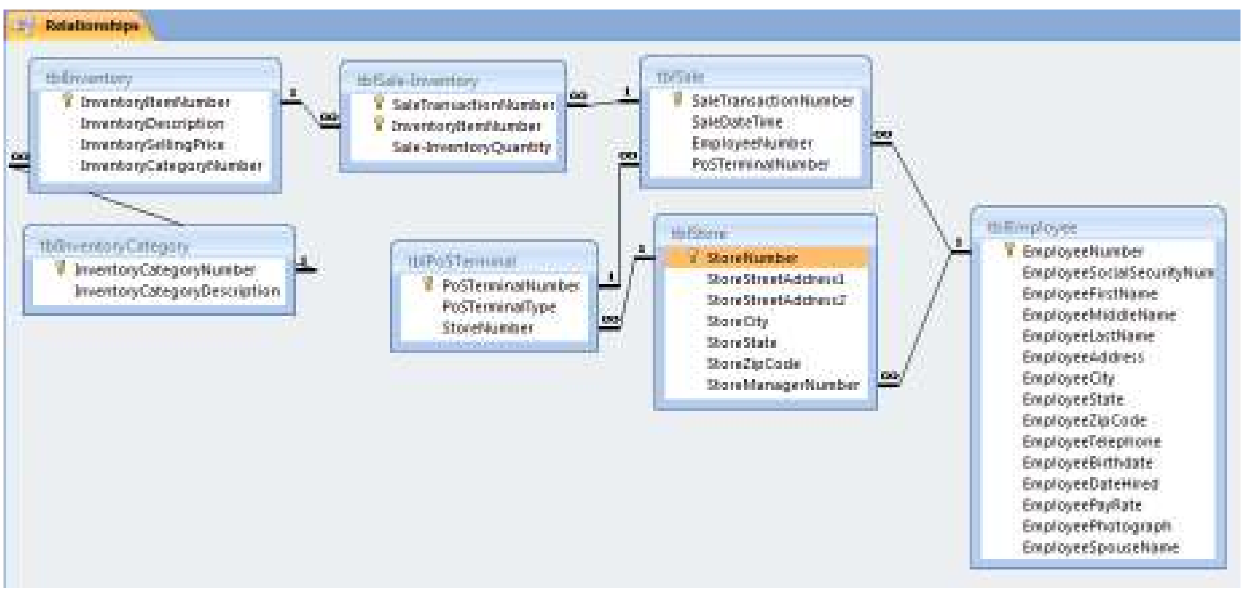
1. One possible solution is shown below in Design view and Datasheet view.



qryPracticeExercise01

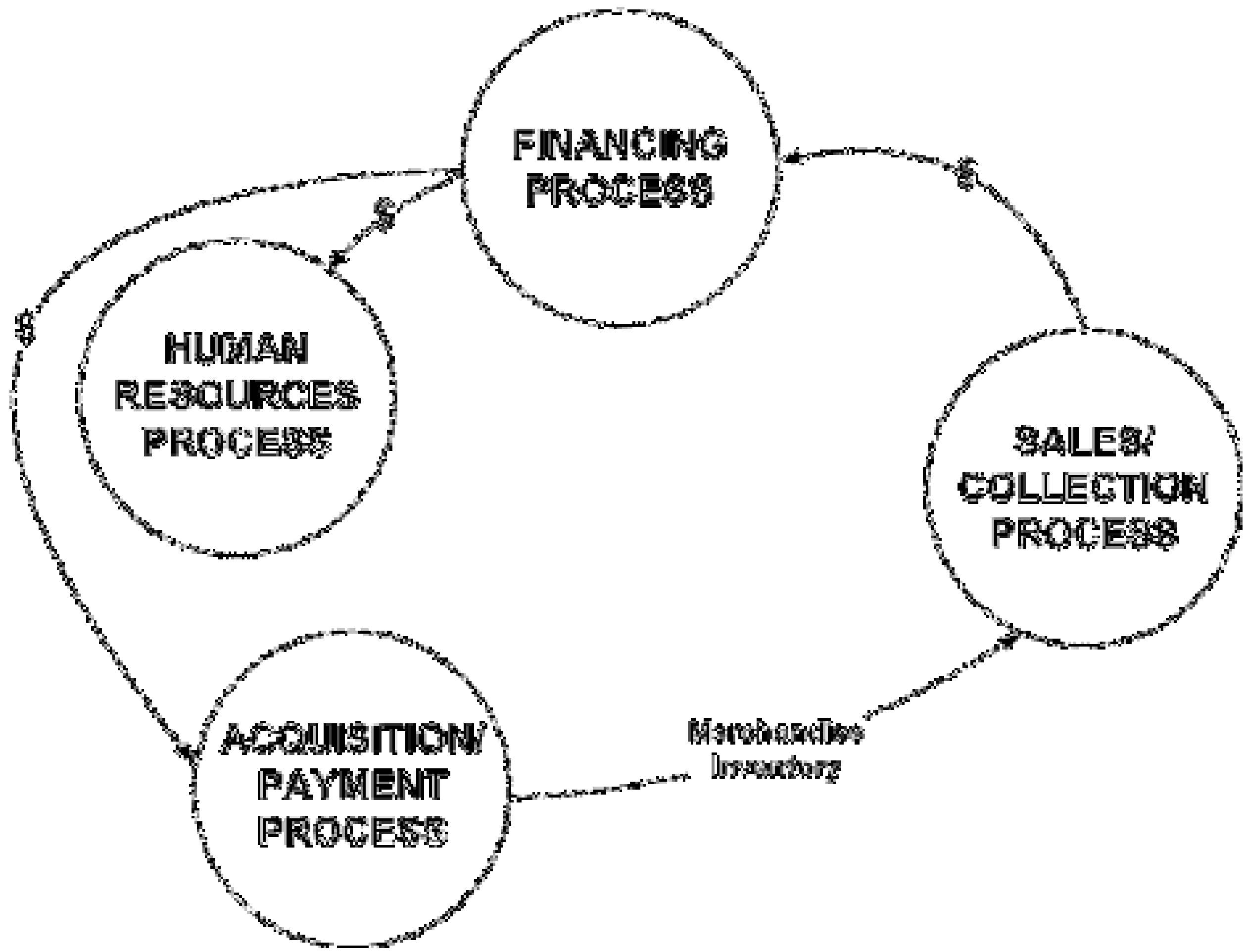
InventoryID	Price	OnHand
1125	$4.50	4,740
1127	$4.50	1,800
1134	$4.50	4,870
1151	$4.50	-10
1172	$4.30	2,550
1180	$4.50	5,340
1191	$4.50	2,220
1196	$4.50	1,640
1208	$4.50	-110
1215	$4.70	3,640
*		

2. One possible solution to this exercise is the database shown in the following figure, which shows a screen capture of an Access Relationships window. In creating this file, we assumed that the grocery store does not track its customers' names or addresses and that it only accepts cash. The tblPoSTerminal stores information about each point-of-sale (PoS) terminal in each store. This database design is a useful example because it illustrates that not all businesses include customer tables in their sales/collection processes.



3. The additional entities would be best modeled in two separate tables, an Office table and a Department table. The primary keys of these two tables would each appear as a foreign key in the Time Worked table.

4. Omit the conversion process bubble and the related resource flow arrows from Figure 2.2. Also, add a resource flow line for merchandise inventory from the acquisition/payment process to the sales/collection process to yield the correct answer. See below.



5. The problem is that age is a value that changes each year. The solution is to store a person's date of birth. That never changes. Follow the general rule "never store any value in a table that can be derived from other values in the table or the external environment." Age is such a value—it can be calculated from date of birth and the current date, which is stored in the computer's clock.

Problems

1. One possible solution is the object qryProblem01, which is included in the Ch02Problem01.mdb database on the Instructor's CD. The figures below show that query in Design view and in Datasheet view.

qryProblem1

tblInventoryDescription

- *
- ItemID
- Name
- BeverageType
- Flavored
- CountryID
- Comments

Field:	ItemID	Name	Flavored	BeverageType
Table:	tblInventoryDescription	tblInventoryDescription	tblInventoryDescription	tblInventoryDescription
Sort:				
Show:	☑	☑	☑	☑
Criteria:			Yes	"c"

qryProblem1

Item	Name	Flavored	BeverageType
182	Columbia Bucaramanga Especial	☑	c
386	Almond Amaretto	☑	c
389	Apricot Cordial	☑	c
392	Chocolate Brandy	☑	c
395	Chocolate Cinnamon	☑	c
398	Chocolate Hazelnut	☑	c
401	Chocolate Macadamia	☑	c
404	Chocolate Mint	☑	c
407	Chocolate Raspberry	☑	c
410	Cinnamon	☑	c
413	Coconut Cream	☑	c
416	Dark Chocolate Almond	☑	c
419	Dutch Chocolate	☑	c
422	French Vanilla	☑	c
425	Hazelnut	☑	c
431	Irish Cream	☑	c
434	Kahlua	☑	c
437	Macadamia	☑	c
440	Seville Orange	☑	c
443	Southern Pecan	☑	c
449	Vanilla Almond	☑	c
452	Vanilla Nut Creme	☑	c
455	Vanilla Nut Fudge	☑	c
* 0		■	

2. The Invoice Line table is a relationship table. All relationship tables have a composite primary key that includes two or more individual fields (the primary key fields of the entity tables that the relationship table joins). In this case, tblInvoiceLine joins the Invoice table and the Inventory table. Thus, the primary key of tblInvoiceLine includes the InvoiceID field and the InventoryID field. The extension is not included in the table because it is a calculated field. A printed invoice would be produced by creating a report based on a query. That query would calculate the extension.

3. The relationship between the Instructor table and a Classes table is one-to-many (1—M). Each instructor teaches many classes (unless he or she is a researcher and teaches only one class per semester—atypical of most instructors). The relationship between Students table and the Classes table is many-to-many (M—M). That is, a given student can take several classes. Any particular class can have several students enrolled in it. The following shows a typical relationship between the Catalog, Classes, Instructors, and Students. Notice a relationship table exists when necessary to convert M—M relationships to two 1—M relationships.