CHAPTER 1

Data Basics

Data comes in many different forms. Whether the data is a personal contact history, a set of academic test scores, a catalog of products and prices, a group of scientific research facts, or a multinational corporation's general ledger entries for the past 20 years, data can be small or large, simple or complex, and summarized or detailed.

Understanding the differences between common database types—flat file databases, nonrelational databases, relational databases, and multidimensional databases—will help you decide whether to use Microsoft Office Excel, Microsoft Office Access, Microsoft SQL Server, or a similar database management system from another computer software manufacturer to enter, store, modify, and analyze your particular data.

1.1 Learn About Flat File Databases

A *flat file database* is a single electronic text file containing a list of data records with one record per line, usually with a newline character separating each data record. Each record contains one or more data fields with each field separated by a character, known as a *delimiter*, such as a comma or a tab character. For example, in a list of personal contacts, each data record contains an individual contact's information: the contact's name, address, and phone number are each a data field.

Flat file databases are ideal for storing simple data values, especially when those values are in data records with varying numbers of fields. However, flat file databases can be tough to enter data into; specifically, they are error-prone when entering multiple data field delimiters.

Flat file database data records and data fields usually are consistent in their definition, layout, and data format, such as the personal contact list described earlier, but this is not strictly required. For example, in a flat file database containing a list of students and their test scores, the first data record could contain a student's name and five numeric test score data fields, while the second data record could contain a student's identification number and seven alphabetic test score data fields.

Quick Start

A flat file database can most easily be represented as an electronic text file with each data record separated usually by a newline character. For each data record, each data field in that data record is separated by a common character such as a comma or a tab character.

How To

To quickly create a flat file database, use one of two ways. The first is the following:

1. Start Microsoft Notepad.
2. Type a series of data records with each data field value separated by a common character such as a comma or a tab character.
3. Press Enter after each data record.
4. Save the file.

The other way is the following:

1. Start Excel.
2. Type a series of data records with each data field in a subsequent worksheet cell.
3. Enter each data record on a subsequent worksheet row.
4. Save the file.

Tip

You should only use flat file databases for the simplest lists of data values. Flat file databases are prone to corruption, especially when two or more users or computer programs are trying to work with the same flat file database at the same time. Flat file databases are also prone to data entry errors. If you miss entering just one delimiter in a flat file database, you increase the probability of a database management system to not be able to correctly open, display, analyze, or store the data values.

Try It

In this exercise, you will open a flat file database in Notepad. Then you will open the same flat file database in Excel to see how Excel presents flat file data in rows and columns on a worksheet:

1. Start Microsoft Notepad.
2. Click File ➤ Open.
3. Browse to and select the ExcelDB_Ch01_01.txt file, and click Open. Notice that each data field is separated by a comma, and each data record is on a separate line.
4. Start Excel.
5. Click Office Button ➤ Open (for Excel 2007) or click File ➤ Open (for Excel 2003). In the Files of Type box, select All Files.
6. Browse to and select the ExcelDB_Ch01_01.txt file, and click Open. The Text Import Wizard appears.
7. Select the Delimited option, and then click Next.

8. Clear the Tab check box, select the Comma check box, and click Finish. Notice that each data field is in a separate worksheet cell, and each data record is on its own row.

9. Quit Excel, and quit Notepad.

1.2 Learn About Nonrelational Databases

The defining characteristics of a *nonrelational database* are that each data table (which is a collection of individual data records) in a nonrelational database is self-describing and self-contained. For example, in a nonrelational database containing a personal contact list, the contact list itself is a single data table; each contact is a data record; each contact's first name is a data field; and each contact's street address is another data field. Furthermore, the data field values are straightforward to understand, and the contact list does not depend on any other data tables to convey each contact's information.

Nonrelational databases are great for storing lists of data values with the following:

- The same number of data fields in each data record.
- Data values and data records that do not depend on other data tables to convey all of the information about each data record.
- Data values that are straightforward to understand.
- Data fields that are organized with similar data values grouped together.

There are two key differences between flat file databases and nonrelational databases. The first key difference is that a flat file database does not need to have the same number of data fields per data record. Nonrelational databases always have the same number of data fields per data record.

The second key difference between flat file databases and nonrelational databases is that flat file databases do not need to contain data field names. Nonrelational databases always contain data field names.

Quick Start

A nonrelational database is simply an electronic file containing the same number of data fields in each data record, and each data field has a name. Similar to a flat file database, you could represent a nonrelational database as a text file containing a set of data records, with each data record separated usually by a newline character. Each data field in a data record is separated by a common character such as a comma or a tab character. Each data record contains the same number of data fields.

How To

To quickly create a nonrelational database, use one of two ways. One way is the following:

1. Start Notepad.

2. Type a series of data field names, with each data field name separated by a common character such as a comma or a tab character, and press Enter.

3. Type a series of data records with each data field value separated by a common character such as a comma or a tab character. Make sure that each data record has the same number of data field values as data field names.
4. Press Enter after each data record.
5. Save the file.

The other way is the following:

1. Start Excel.
2. In the first row of a worksheet, type a series of data field names, with each data field name in a subsequent worksheet cell.
3. In the second and subsequent rows, type a series of data field values with a data field value or a null value for each data field name.
4. Enter each data record on a subsequent worksheet row.
5. Save the file.

Tip

A data field in a nonrelational database that contains no data value for a given data record is commonly known as a *null value* or a *null field*. Null values are commonly expressed as a blank value, the value Null, or the value N/A (for not applicable). Note that the value zero (0) is never used to convey a null value.

For most data entry, storage, and analysis tasks, Excel handles flat file databases and nonrelational databases the same.

Try It

In this exercise, you will open a nonrelational database in Notepad. Then you will open the same nonrelational database in Excel to see how Excel presents the data in rows and columns on a worksheet:

1. Start Notepad.
2. Click File ➤ Open.
3. Browse to and select the ExcelDB_Ch01_02.txt file, and click Open. Notice that the first line contains data field names; each data field is separated by a comma; each data record is on a separate line; and there are the same number of data field values for each data record.
4. Start Excel.
5. Click Office Button ➤ Open (for Excel 2007) or click File ➤ Open (for Excel 2003). In the Files of Type box, select All Files.
6. Browse to and select the ExcelDB_Ch01_02.txt file, and click Open. The Text Import Wizard appears.

7. Select the Delimited option, and click Next.
8. Clear the Tab check box, select the Comma check box, and click Finish. Notice that each data field is in a separate worksheet cell; each data record is on its own row; and there are the same number of data field values for each row.

Tip To see all of the data field names and data field values, click the Select All button (the blank button in the upper left corner of the worksheet), and click Home ➤ (Cells) Format ➤ AutoFit Column Width (for Excel 2007) or Format ➤ Column ➤ AutoFit Selection (for Excel 2003).

---

9. Quit Excel, and quit Notepad.

1.3 Learn About Relational Databases

Similar to nonrelational databases discussed in the previous section, *relational databases* store data records in two or more data tables. However, relational databases are different than nonrelational databases in one key aspect: the data tables rely on each other to capture all of the facts and figures in the database. For example, in a nonrelational database containing customer sales history, one data table contains all of the customers' names and addresses and all of the sales transactions for all of the customers. In contrast, in a relational database containing customer sales history, one data table would contain the customers' names and addresses, while another data table would contain all of the sales transactions for all of the customers.

You should consider using relational databases for all but the simplest of data lists. Very large flat file and nonrelational databases can be slow to open, tough to search in for specific data records, and prone to data-entry errors and data corruption.

There are two main benefits to using relational databases vs. nonrelational databases. The first benefit of using relational databases is the efficient use of database space. Using the example of the nonrelational database in the preceding section, there would be a lot of repeated customer names and addresses and therefore increased wasted space. The second benefit of using relational databases is the reduction of data-entry errors. Duplicating data can increase the probability of data-entry errors every time you retype the same customer names and addresses. Once you remove the repeated customer names and addresses to a separate data table in a relational database, you can update the customer names and addresses in just one table.

To declare relationships among data tables and cross-reference related data records in separate data tables to each other in a relational database, you use *primary keys* and *foreign keys*. A primary key is a data field containing a unique identifier—such as a sequential number, a part number, a customer ID, or a Social Security number—applied to each data record in the main table, also known as the *primary-key data table*. A foreign key then is a data field in the related table, also known as the *foreign-key data table*, containing the unique identifier from the related data record in the primary-key data table. For example, in the relational database example in the preceding section, you could assign each customer in the customer data table a unique ID number, and include the customer's unique ID number in each data record in the sales transactions data table for that customer.

Quick Start

To create a relational database, create two or more data tables, and then enter data records into each data table. Make sure that each data table contains a primary-key data field and that each data record in that data table contains a unique identifier in the primary-key data field. Also, for each related data table, create a foreign-key data field, and make sure that each data record in the related data table contains a primary-key data value from the related record in the primary-key data table.

How To

To create a relational database in Excel, do the following:

1. Start Excel.
2. Using one worksheet per data table, enter data records into each table.
3. Make sure that each worksheet contains a primary-key data field.
4. Make sure that for each worksheet, each data record in that worksheet has a primary-key data value in the primary-key data field that is unique to that worksheet.
5. Make sure that for each worksheet with data records related to the primary-key data table worksheet, the related worksheet contains a foreign-key field.
6. Make sure that each data record in the related worksheet contains a primary-key data value in the foreign-key data field, with that primary-key data value taken from the related record in the primary-key data table worksheet.
7. Save the file.

Tip

Foreign-key data tables should always also contain a primary-key data field. For example, a customer data table could have a related sales transactions data table, which in turn could have a related sales products data table. In this case, the sales transactions data table would need a foreign-key data field to cross-reference unique customers to sales transactions, and the sales transactions data table would also need a primary-key data field to relate unique sales transactions to unique sales products. (Of course, the customer data table would also need a primary-key data field to uniquely identify each customer, and the sales products data table would also need a primary-key data field to uniquely identify each sales product.)

Try It

In this exercise, you will examine a relational database in Excel. You will then use Access to import the relational data, examine the data in Access, define data table relationships, and examine related data:

1. Start Excel.
2. Click Office Button ➤ Open (for Excel 2007) or click File ➤ Open (for Excel 2003).

3. Browse to and select the ExcelDB_Ch01_03.xls file, and click Open. Notice that there are five worksheets in this workbook, one worksheet each for the Orders, Line Items, Suppliers, Products, and Salespeople data tables. In each worksheet, the primary key field ends in "PK," and any foreign key fields end in "FK."

4. Close the workbook.

Now, import the workbook data into Access.
For Access 2007, do the following:

1. Start Access.

2. Click Office Button ➤ New.

3. In the Blank Database pane, in the File Name box, type any name that's easy for you to remember for the database, click the Browse for a Location to Put Your Database icon and select a location for the database, and then click Create.

■ **Note** You may need to scroll down the screen to find the Create button if the Create button is not visible under the File Name box.

---

4. Click External Data ➤ (Import) Excel.

5. Click Browse, browse to and select the ExcelDB_Ch01_03.xls file, click Open, and click OK.

6. Click the Show Worksheets option, select Orders in the list of available worksheets, and then click Next.

7. Select the First Row Contains Column Headings check box, and then click Next.

8. In the Indexed list, select Yes (No Duplicates), and then click Next.

9. Select the Choose My Own Primary Key option, select Order_ID_PK, and then click Next.

10. Click Finish, and then click Close. The Orders table is imported into the Access database.

11. Repeat steps 4 through 10 to import the Line Items, Suppliers, Products, and Salespeople worksheets into the Access database. Be sure to substitute in step 9 the values Line_ID_PK, Supplier_ID_PK, Product_ID_PK, and Salesperson_ID_PK for Order_ID_PK as appropriate. You can check your results against the imported worksheets in the finished ExcelDB_Ch01_03.mdb database file.

12. Open each of the tables in Access to ensure that the data in the Orders, Line Items, Suppliers, Products, and Salespeople data tables match the data in the Excel workbook. You can check your results against the imported worksheets in the finished ExcelDB_Ch01_03.mdb database file if needed.

For Access 2003, do the following:

1. Start Access.
2. Click File ➤ New.
3. In the New File task pane, click Blank Database, type any name that's easy for you to remember for the database in the File Name box, browse to a location to put your database, and then click Create.
4. Click File ➤ Get External Data ➤ Import.
5. In the Files of Type list, select Microsoft Excel.
6. Browse to and select the ExcelDB_Ch01_03.xls file, and click Import.
7. Select the Show Worksheets option, select Orders in the list of available worksheets, and then click Next.
8. With the First Row Contains Column Headings check box selected, click Next.
9. With the In a New Table option selected, click Next.
10. In the Indexed list, select Yes (No Duplicates), and click Next.
11. Select the Choose My Own Primary Key option, select Order_ID_PK, and click Next.
12. Click Finish, and click OK. The Orders table is imported into the Access database.
13. Repeat steps 4 through 12 to import the Line Items, Suppliers, Products, and Salespeople worksheets into the Access database. Be sure to substitute in step 11 the values Line_ID_PK, Supplier_ID_PK, Product_ID_PK, and Salesperson_ID_PK for Order_ID_PK as appropriate. You can check your results against the imported worksheets in the finished ExcelDB_Ch01_03.mdb database file.
14. Open each of the tables in Access to ensure that the data in the Orders, Line Items, Suppliers, Products, and Salespeople data tables match the data in the Excel workbook. You can check your results against the imported worksheets in the finished ExcelDB_Ch01_03.mdb database file if needed.

Next, create relationships among the data tables in Access:

1. For Access 2007, click Database Tools ➤ (Show/Hide) Relationships. For Access 2003, click Tools ➤ Relationships.
2. On the Show Table dialog box's Tables tab, with the Line Items data table selected, click Add. Repeat this step for the Orders, Products, Salespeople, and Suppliers data tables. Then click Close.
3. In the Orders data table, drag the Order_ID_PK data field to the Line Items data table's Order_ID_FK data field.

■ **Note** Be sure to close all of the open data tables in Access before you complete the preceding step.

4. In the Edit Relationships dialog box, select the Enforce Referential Integrity check box, and then click Create.

■ **Note** Selecting the Enforce Referential Integrity check box ensures that Access will prevent you from deleting a data record in the primary data table when there are matching data records in a related data table. This prevents you from having "stranded" or "orphaned" data in related data tables.

5. Repeat steps 3 and 4 for the following data fields:
   - In the Products data table, drag the Product_ID_PK data field to the Line Items data table's Product_ID_FK data field.
   - In the Salespeople data table, drag the Salesperson_ID_PK data field to the Orders data table's Salesperson_ID_FK data field.
   - In the Suppliers data table, drag the Supplier_ID_PK data field to the Products data table's Supplier_ID_FK data field.
   - You can check your results against the finished ExcelDB_Ch01_03.mdb database file.
6. Click Office Button ➤ Save (for Excel 2007) or File ➤ Save (for Excel 2003).
7. Close the Relationships window.

Now that you have data table relationships defined, drill down into one of the supplier's sales order details in Access.

1. Open the Suppliers data table.
2. Click the plus sign symbol next to the Acme data row.
3. Click the plus sign symbols next to the two products that are displayed to discover how many units were ordered on which orders.
4. Quit Access, and quit Excel.

1.4 Normalize Data

Relational databases work best when data is normalized. When you normalize your data, you eliminate redundant data to help protect your data against data entry errors. You also ensure that the information in each data table is correctly linked so that you can properly cross-reference related data.

You normalize data when you have a lot of repetitive data in one or more data tables and you want to restructure the data to reduce data entry errors and possibly reduce data storage requirements.

To normalize data, you should follow a set of well-established rules called *normal forms*. There are three common normal forms. There are also several less common normal forms that are beyond the scope of this book.

The general strategies underlying the three common normal forms are the following:

- Eliminate repeating data in rows or data records.
- Eliminate repeating data in columns or data fields, moving the repeated data to other data tables.
- Use primary keys and foreign keys to cross-reference related data records among data tables.

For example, examine the following nonnormalized data in Table 1-1.

Table 1-1. *Nonnormalized Weather Data for Three United States Cities*

City, State	Date 1	High	Low	Air Quality	Date 2	High	Low	Air Quality
Portland, Oregon	15-Feb	47	30	Moderate	16-Feb	45	26	Moderate
Portland, Oregon	17-Feb	33	23	Good	18-Feb	39	27	Good
Salem, Oregon	15-Feb	47	27	Moderate	16-Feb	44	23	Moderate
Salem, Oregon	17-Feb	31	22	Good	18-Feb	39	23	Good
Spokane, Washington	15-Feb	35	18	Good	16-Feb	23	2	Good
Spokane, Washington	17-Feb	20	10	Good	18-Feb	32	14	Good

Notice the following facts in the preceding data table:

- The cities and states are contained in the same data field, with several duplicate cities and states listed.
- The date, high temperature, low temperature, and air quality data fields are presented in a peculiar manner: the weather for four dates is presented in more than four data records; and three city and state combinations are presented in more than three records.
- Many air quality data field values are repeated.

By moving repeating data to other data tables and linking the data tables together through primary keys and foreign keys, you could present the data in Tables 1-2 through 1-7.

Table 1-2. *Cities Data Table for Normalized Weather Data from Table 1-1*

City_ID_PK	City
1	Portland
2	Salem
3	Spokane

Table 1-3. *States Data Table for Normalized Weather Data from Table 1-1*

State_ID_PK	State
1	Oregon
2	Washington

Table 1-4. *Cities States Data Table for Normalized Weather Data from Table 1-1*

City_State_ID_PK	City_ID_FK	State_ID_FK
1	1	1
2	2	1
3	3	2

Table 1-5. *Dates Data Table for Normalized Weather Data from Table 1-1*

Date_ID_PK	Date
1	15-Feb
2	16-Feb
3	17-Feb
4	18-Feb

Table 1-6. *Air Qualities Data Table for Normalized Weather Data from Table 1-1*

Air_Quality_ID_PK	Air_Quality
1	Moderate
2	Good

Table 1-7. *Weather Data Data Table for Normalized Weather Data from Table 1-1*

Data_Record_ID_PK	Date_ID_FK	City_State_ID_FK	High	Low	Air_Quality_ID_FK
1	1	1	47	30	1
2	2	1	45	26	1
3	3	1	33	23	2
4	4	1	39	27	2
5	1	2	47	27	1
6	2	2	44	23	1
7	3	2	31	22	2
8	4	2	39	23	2
9	1	3	35	18	2
10	2	3	23	2	2
11	3	3	20	10	2
12	4	3	32	14	2

Normalizing this data results in the following benefits:

- The Cities, States, and Cities States data tables are extendable to allow a city with the same name to exist in multiple states.
- If a city or state changes its name, you only need to change a record in the Cities or States data table.
- If the representation of a date needs to change (for example, changing 15-Feb to 02/15 or 15/02), you only need to change data records in the Dates data table.
- If the air quality categories change, you only need to change data records in the Air Qualities data table.

As an added side benefit, sorting and averaging weather data is a bit more straightforward in the normalized Weather Data data table. In the nonnormalized data for example, averaging high temperatures for cities in Oregon for February 15 is more complicated: first you must filter for all rows where Oregon is somewhere in the City, State data field, then you must somehow collect all of the High data field values together where the corresponding Date 1 or Date 2 data field is 15-Feb (which is tough for many database management systems to do automatically), then you calculate the average high temperature. In the normalized Weather Data data table, you filter for all rows where the Date_ID_FK data value contains a matching data value in the Dates data table corresponding to 15-Feb and where the City_State_ID_FK data value contains a matching value in the Cities States and States data tables corresponding to Oregon; then you average the values in the High data field.

Quick Start

To normalize repetitive data, you eliminate the repeating data in data records and data fields, moving the repeating data to other data tables. You then use primary keys and foreign keys to cross-reference related data records among those data tables.

How To

To normalize data in one or more existing data tables, do the following:

1. Identify data fields with repeating data values or multipart data values (for example, contact name and address data values or product name and manufacturer data values contained in the same data field). Break these data values into multiple data fields (for example, separate data fields for name, address, product name, or manufacturer data values).

2. Group data fields with related data values into separate data tables (for example, a data table for contacts, a data table for products, or a data table for manufacturers).

3. Eliminate repeating data values in each data table (for example, a repeated address or a repeated product name).

4. Assign a primary key data field to each data table and a unique identifier for each data record in that data table (for example, a unique contact identification number or a unique product part number).

5. Add foreign key data fields as needed to cross-reference related data records contained in multiple data tables (for example, foreign key data fields describing the relationships between products and manufacturers, cross-referencing primary key data values in the separate product and manufacturer data tables).

6. Create additional data tables and use foreign keys as needed to store data records containing unique facts and figures (for example, a product sales transaction data table containing individual sales transaction details, cross-referencing primary key data values in the product/manufacturer data table).

Tip

A *one-to-many relationship* between two data tables is the most common type of relationship. A one-to-many relationship exists when a data record in data table A can have many matching data records in another data table B, but a data record in data table B has only one matching data record in data table A. For example, a sales order in one data table can have many matching sales line items in another data table, but each sales line item matches only one sales order.

A less frequent but still common type of relationship, a *many-to-many relationship*, exists between two data tables when a data record in table A can have many matching records in data table B, and a record in data table B can have many matching data records in data table A. A many-to-many relationship is made possible by creating a third table, called a *junction table*, that contains foreign keys from both data tables A and B. A many-to-many relationship is really two one-to-many relationships described by a third data table. For example, a sales order in one data table can have many matching product items in another data table, and each product item can appear in many different sales orders. A third data table is used to describe this complex relationship, matching sales orders to product items and product items to sales orders.

A very uncommon type of relationship, a *one-to-one relationship*, exists between two data tables when each data record in data table A can have only one matching data record in data table B, and each data record in data table B can have only one matching data record in data table A. This type of relationship is uncommon because this type of data is best described in a single data table. For example, a customer in one data table has a unique identification number in another data table, and each unique identification number belongs to only one customer. In this example, you may have needed to separate this information into two data tables for some type of security or privacy policy mandated by your organization.

Try It

In this exercise, you will practice using Excel and Access to normalize some nonnormalized data:

1. Start Excel.
2. Click Office Button ➤ Open (for Excel 2007) or click File ➤ Open (for Excel 2003).
3. Browse to and select the ExcelDB_Ch01_04.xls file, and click Open. Notice that there are six worksheets in this workbook, one worksheet with some nonnormalized data in it, and one worksheet each labeled Orders, Line Items, Suppliers, Products, and Salespeople.

You can use these worksheets to manually normalize the nonnormalized data, comparing your results to the ExcelDB_Ch01_04.mdb file. Or you can follow the remaining steps to import the nonnormalized data into Access, normalize the data, and use Excel to import the normalized data.

For Access 2007, do the following:

1. Start Access.
2. Click Office Button ➤ New.
3. In the Blank Database pane, click Create.
4. Click External Data ➤ (Import) Excel.
5. Click Browse, browse to and select the ExcelDB_Ch01_04.xls file, click Open, and click OK.
6. Select the Show Worksheets option, select Nonnormalized Data in the list of worksheets, and then click Next.
7. Select the First Row Contains Column Headings check box, and then click Next.
8. In the Indexed list, select Yes (Duplicates OK), and click Next.
9. With the Let Access Add Primary Key option selected, click Next.
10. Click Finish, and click Close. The Nonnormalized Data data table is imported into the Access database.

For Access 2003, do the following:

1. Start Access.
2. Click File ➤ New.
3. In the New File task pane, click Blank Database.

4. Click Create.
5. Click File ➤ Get External Data ➤ Import.
6. In the Files of Type list, select Microsoft Excel.
7. Browse to and select the ExcelDB_Ch01_04.xls file, and click Import.
8. Click the Show Worksheets option, select Non-Normalized Data in the list of worksheets, and then click Next.
9. Select the First Row Contains Column Headings check box, and click Next.
10. With the In a New Table option selected, click Next.
11. In the Indexed list, with the Yes (Duplicates OK) item selected, click Next.
12. With the Let Access Add Primary Key option selected, click Next.
13. Click Finish, and click OK. The Nonnormalized Data data table is imported into the Access database.

Next, use the Table Analyzer Wizard to help you normalize the data in the Nonnormalized Data data table:

1. With the Nonnormalized Data data table selected (but not opened), for Access 2007, click Database Tools ➤ (Analyze) Analyze Table. For Access 2003, click Tools ➤ Analyze ➤ Table.
2. Click Next three times.
3. Click the No, I Want to Decide option, and then click Next.
4. Drag the Order_ID data field from the Table2 data table (for Access 2007) or Table1 data table (for Access 2003) to a blank area of the workspace. Type **Orders**, and click OK.
5. Drag the Salesperson_Name data field from the Table2 data table (for Access 2007) or Table1 data table (for Access 2003) to a blank area of the workspace. Type **Salespeople**, and click OK.
6. Drag the Supplier_Name data field from the Table2 data table (for Access 2007) or Table1 data table (for Access 2003) to a blank area of the workspace. Type **Suppliers**, and click OK.
7. Drag the Product_Description data field from the Table2 data table (for Access 2007) or Table1 data table (for Access 2003) to a blank area of the workspace. Type **Products**, and click OK.
8. Drag the Unit_Description data field from the Table2 data table (for Access 2007) or Table1 data table (for Access 2003) to underneath the Product_Description data field in the Products data table.
9. Drag the Price_Per_Unit data field from the Table2 data table (for Access 2007) or Table1 data table (for Access 2003) to underneath the Unit_Description data field in the Products data table.

10. Click the title bar of the Table2 data table (for Access 2007) or Table1 data table (for Access 2003), click the Rename Table button, type **Line Items**, and click OK. Compare your results to Figure 1-1.

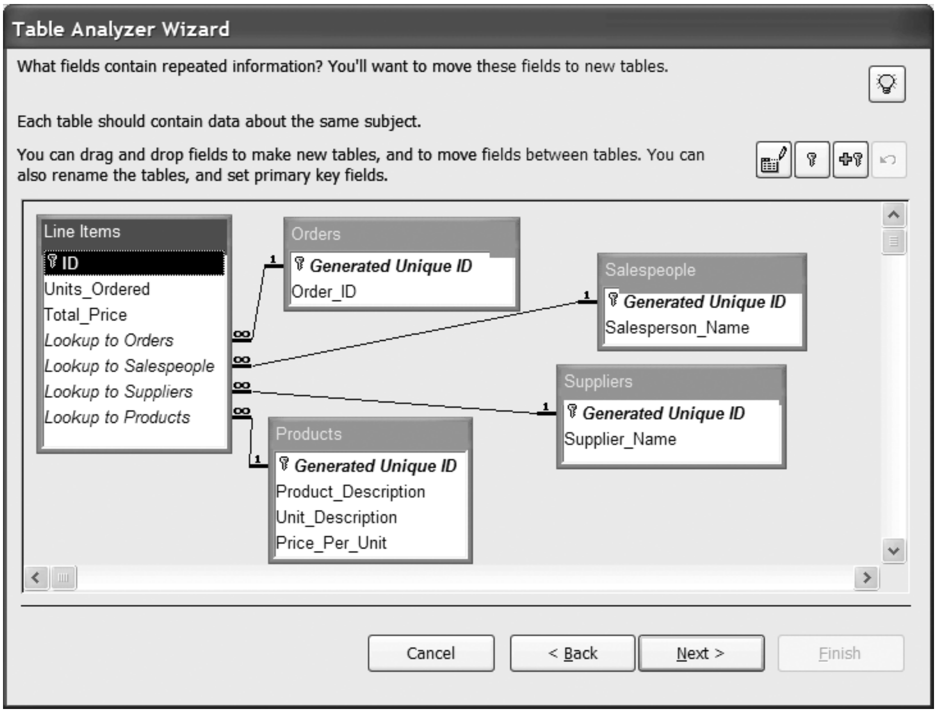


Table Analyzer Wizard

What fields contain repeated information? You'll want to move these fields to new tables.

Each table should contain data about the same subject.

You can drag and drop fields to make new tables, and to move fields between tables. You can also rename the tables, and set primary key fields.

Figure 1-1. *The completed table design in the Table Analyzer Wizard*

11. Click Next five times.

Note Each time you click Next, you will be asked if you really want to move on without changing anything. In each case click Yes.

12. On the final page of the Table Analyzer Wizard, with the Yes, Create the Query option selected, click Finish.

13. Click OK. (For Access 2003 only, a query simulating the contents of the original Non-normalized Data data table appears. Click File ➤ Close to return to the Database Objects window.)

14. Open and explore the contents of the normalized Line Items, Orders, Products, Salespeople, and Suppliers tables.

1.5 Learn About Multidimensional Databases

Microsoft Office Excel 2003 can display at most 65,536 data records or 256 data fields on a single worksheet. (Excel 2007 supports 1 million data records and 16,000 data fields on a single worksheet.) What happens if your data exceeds any of these limits? You have a few options, specifically the following:

- You could break up the data into a set of smaller data tables, although this approach could be very difficult to set up and maintain in Excel. Also, multiple data cross-reference operations could be a big drain on your computer's resources.
- You could use Microsoft Office Access 2007 or Access 2003 to store and manage your data, although with Access 2007 and 2003 you are still limited to 255 data fields in a single table. You are also limited to 4,000 characters in a data record for Access 2007 (2,000 characters for Access 2003), excluding a few special types of data fields.
- You could use other database management systems that are more robust for very large databases compared to Excel—for example, Microsoft SQL Server. But these systems are more expensive, are harder to learn and maintain, require greater computing resources, and lack most of Excel's great data analysis features.

There is, however, one more approach for representing in Excel large amounts of data that exceed Excel's limits, and that approach is to convert the data into a *multidimensional database*.

You should consider using a multidimensional database when you have a large amount of data that exceeds Excel's display limits, or when you have a large database for which you only want to work with summarized data and not necessarily the individual data values themselves.

A multidimensional database is a set of data records that summarize the most important facts and figures in a large database. The term *multidimensional* comes from the fact that the summarized facts and figures can be cross-referenced along several dimensions. Here are a few of the key terms you should know about when working with multidimensional data:

- *Dimensions* are categories or groupings of similar facts and figures such as time, geography, products and services, or organizations.
- Dimensions can be further broken down into *levels*. A time-oriented dimension could consist of years, seasons, months, and weeks. A geographical-oriented dimension could consist of continents, countries or regions, and states or provinces.
- Levels consist of *members*. A year level could contain the members 2004, 2005, 2006, and 2007. A country level could contain the members France, Germany, and Italy.
- *Measures* contain the summarized data values. Measures can be summarized by member, level, or dimension, depending on how much detail you are interested in working with.
- Dimensions, levels, members, and measures are stored in electronic files called *cube files*. Just as a physical cube has three dimensions, a multidimensional data cube file also contains dimensions.

Note Cube files are not cubes in the strictly geometrical sense because they are not limited to three dimensions. However, the term *cube file* in this context is well understood and defined in the discipline of multidimensional data analysis.

Because a cube file contains various combinations of measures for members, levels, and dimensions summarized in advance, you can retrieve the summarized data very quickly. In fact, because cube files only contain summarized representations of the data records' key facts and figures, cube files are smaller in size than the data upon which they are based. Because of their smaller size, cube files can use fewer computing resources to work with.

You can use Excel to both create and work with cube files.

Quick Start

To create and save a cube file in Excel 2003, do the following:

Note Excel 2007 does not support creating a cube file using MS Query. This feature has been removed from Excel 2007 due to technical issues. You can, however, create cube files based on cubes that exist in Microsoft SQL Server Analysis Services. You can also create cube files from relational databases using Microsoft SQL Server Analysis Services.

1. Start Excel.
2. Click Data ➤ Import External Data ➤ New Database Query.
3. With the Use the Query Wizard to Create/Edit Queries check box selected, on the Databases tab, click one of the items in the list to create a connection to an existing external data source (such as a dBASE file, an Excel workbook, or an Access database), and then click OK.
4. Follow the steps in the Query Wizard.
5. In the Query Wizard – Finish page, select the Create an OLAP Cube from this query option, and click Finish.
6. Complete the steps in the OLAP Cube Wizard to finish creating the cube file.

To open and work with a cube file in Excel 2007, do the following:

1. Click Data ➤ (Get External Data) From Other Sources ➤ From Microsoft Query.
2. With the Use the Query Wizard to Create/Edit Queries check box selected and the OLAP Cubes tab selected, select the name of a cube file in the list (or click <New Data Source>, click OK, follow the steps in the Create New Data Source dialog box, click OK, and then select the name of the new data source), and then click OK. The Import Data dialog box appears.

3. Click OK.
4. Create a PivotTable using the PivotTable Field List pane.

To open a cube file in Excel 2003, do the following:

1. Click File ➤ Open.
2. In the Files of Type list, select All Data Sources.
3. Browse to and select a file with the .cub file extension.
4. Click Open.
5. Create a PivotTable using the PivotTable Field List pane.

Note For more information on creating and working with PivotTables, see Chapter 6.

How To

To create a cube file in Excel 2003, do the following:

1. Start Excel.
2. Click Data ➤ Import External Data ➤ New Database Query.
3. With the Use the Query Wizard to Create/Edit Queries check box selected and the Databases tab selected, click one of the items in the list to create a connection to an existing external data source (such as a dBASE file, an Excel workbook, or an Access database), and then click OK.
4. Follow the steps in the Query Wizard.
5. In the Query Wizard – Finish page, select the Create an OLAP Cube from This Query option, and click Finish. The Welcome to the OLAP Cube Wizard page or the OLAP Cube Wizard Step 1 of 3 page appears.
6. If the Welcome to the OLAP Cube Wizard page is displayed, click Next.
7. Complete the steps in the OLAP Cube Wizard to finish creating the cube file.

To open and work with a cube file in Excel 2007, do the following:

1. Click Data ➤ (Get External Data) from Other Sources ➤ From Microsoft Query.
2. With the Use the Query Wizard to Create/Edit Queries check box selected and the OLAP Cubes tab selected, select the name of a cube file in the list (or click <New Data Source>, click OK, follow the steps in the Create New Data Source dialog box, click OK, and then click the name of the new data source), and then click OK. The Import Data dialog box appears.

3. Click OK.
4. Create a PivotTable using the PivotTable Field List pane.

To open a cube file in Excel 2003, do the following:

1. Click File ➤ Open.
2. In the Files of Type list, select All Data Sources.
3. Browse to and select a file with the .cub file extension.
4. Click Open.
5. Create a PivotTable using the PivotTable Field List pane.

Alternatively, for Excel 2003, do the following:

1. Start Excel.
2. Click Data ➤ Import External Data ➤ New Database Query.
3. With the Use the Query Wizard to Create/Edit Queries check box selected and the OLAP Cubes tab selected, select the cube file's name (or click Browse, browse to and select the cube file, and click Open), and click OK. The PivotTable and PivotChart Wizard – Step 3 of 3 dialog box appears.
4. Click Finish.
5. Create a PivotTable using the PivotTable Field List pane.

Tip

To learn more about multidimensional databases, cubes, and a multidimensional data management methodology called *online analytical processing* (OLAP), see Chapter 6, "Analyzing Multidimensional Data with PivotTables," in my book *A Complete Guide to PivotTables: A Visual Approach* (Apress, 2004).

Try It

In this exercise you will use Excel 2003 to create a cube file. Later, you will use Excel 2007 or Excel 2003 to display the cube file's data in Excel.

Using Excel 2003, connect to the data that you will use to create the cube file:

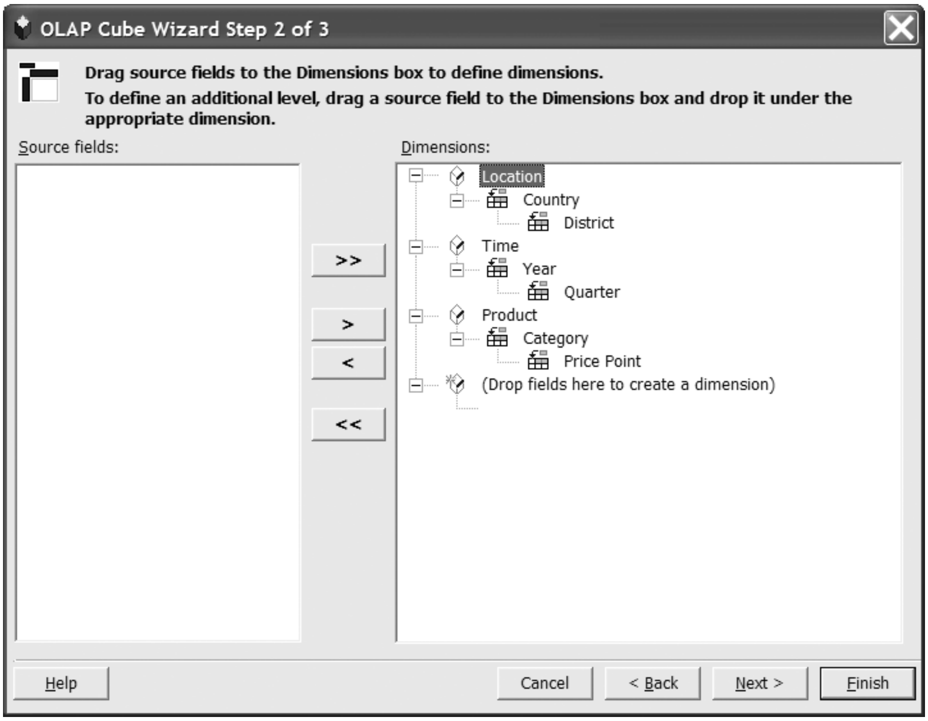
1. Start Excel.
2. Click Data ➤ Import External Data ➤ New Database Query.

3. With the Use the Query Wizard to Create/Edit Queries check box selected and the Databases tab selected, click Excel Files, and click OK.
4. Browse to and select the ExcelDB_Ch01_05.xls file, and click OK.
5. Click Options.
6. Select the System Tables check box, and click OK.
7. In the Available Tables and Columns list, click Data$, click the right arrow (➤) button, and click Next.
8. Click Next two more times.
9. Select the Create an OLAP Cube from This Query option, and click Finish. The Welcome to the OLAP Cube Wizard page or the OLAP Cube Wizard Step 1 of 3 page appears.

Next, use Excel 2003 to create the cube file:

1. If the Welcome to the OLAP Cube Wizard page is displayed, click Next.
2. Clear the Year and Quarter check boxes, and click Next.
3. In the Source Fields list, click Country, and click the right arrow (➤) button.
4. In the Dimensions list, right-click the Country dimension, click Rename, type **Location**, and press Enter.
5. Drag District from the Source Fields list to Country in the Dimensions list.
6. In the Source Fields list, click Year, and click the right arrow button.
7. In the Dimensions list, right-click the Year dimension, click Rename, type **Time**, and press Enter.
8. Drag Quarter from the Source Fields list to Year in the Dimensions list.
9. In the Source Fields list, click Category, and click the right arrow button.
10. In the Dimensions list, right-click the Category dimension, click Rename, type **Product**, and press Enter.
11. Drag Price Point from the Source Fields list to Category in the Dimensions list.

    In the Dimensions list, you should have a Location dimension with a Country member and a District member; a Time dimension with a Year member and a Quarter member; and a Product dimension with a Category member and a Price Point member. Compare your results with Figure 1-2.


OLAP Cube Wizard Step 2 of 3

Drag source fields to the Dimensions box to define dimensions.
To define an additional level, drag a source field to the Dimensions box and drop it under the appropriate dimension.

Source fields:

Dimensions:

- Location
  - Country
    - District
- Time
  - Year
    - Quarter
- Product
  - Category
    - Price Point
- (Drop fields here to create a dimension)

Figure 1-2. *The completed OLAP Wizard Step 2 of 3 page*

12. Click Next.
13. With the Save a Cube File Containing All Data for the Cube option selected, click Browse.
14. In the File Name list, type **ExcelDB_Ch01_05.cub**, and click Save.
15. Click Finish.
16. If the Save As dialog box appears to save the query, type **Excel_Ch01_05** in the File Name box, and click Save. The PivotTable and PivotChart Wizard — Step 3 of 3 dialog box appears.

Finally, display the cube file's data in Excel in 2003:

1. With the PivotTable and PivotChart Wizard — Step 3 of 3 dialog box displayed, click Finish.

   In the PivotTable Field List pane, do the following:

2. Click Location, select Page Area in the Add To list, and click Add To.
3. Click Product, select Row Area in the Add To list, and click Add To.

4. Click Time, select Column Area in the Add To list, and click Add To.
5. Click Sum of Units Sold, select Data Area in the Add To list, and click Add To.

If you are using Excel 2007, connect to the ExcelDB_Ch01_05.cub file included in the Source Code/Download section of the Apress web site, `http://www.apress.com`, and use the PivotTable Field List pane to display the cube file's data, as follows:

1. Start Excel.
2. Click Data ➤ (Get External Data) From Other Sources ➤ From Microsoft Query.
3. With the OLAP Cubes tab selected, click <New Data Source>, and click OK.
4. In the What Name Do You Want to Give Your Data Source box, type **ExcelDB_Ch01_05**.
5. In the Select an OLAP Provider for the Database You Want to Access list, select Microsoft OLE DB Provider for OLAP Services 8.0.
6. Click Connect. The Multidimensional Connection dialog box appears.
7. Click the Cube File option.
8. Next to the File box, click the ellipsis (. . .) button.
9. Browse to and select the ExcelDB_Ch01_05.cub file, click Open.
10. Click Finish, and click OK.
11. In the Choose Data Source dialog box on the OLAP Cubes tab, select ExcelDB_Ch01_05.
12. Click OK. The Import Data dialog box appears.
13. Click OK. The PivotTable Field List pane appears.
14. In the PivotTable Field List pane, select the Sum of Units Sold, Location, Product, and Time check boxes.
15. Drag the Location icon from the Row Labels box to the Report Filter box.
16. Drag the Time icon from the Row Labels box to the Column Labels box.

1.6 Choose the Right Database Product

This chapter concludes with a list of considerations to help you decide whether to use Excel, Access, or SQL Server to address your specific data management needs. You may use Excel for some sets of data, while you may use Access or SQL Server for others.

Use Excel when

- You have a relatively small amount of data.
- Your data is stored in one data table or a relatively small number of tables, and your data tables do not have many complex relationships defined.

- You want to execute data calculations or formulas, perform statistical data analysis, or perform other unique computations on data for which Excel is particularly suited.
- You want to quickly print data without a lot of additional formatting work.
- You don't have any special requirements for data synchronization over multiple remote locations, advanced data backup and restore needs, or ongoing data logging and audit tracking requirements.
- You want to present a simple snapshot of your data over the Web.

Use Access when

- Your data exceeds Excel's operating limits.
- You have more than a few people, but not more than say a couple dozen people, who need simultaneous access to the data.
- You need to synchronize data over a few remote locations.
- You want to take advantage of more robust data entry, querying, and reporting solution options.
- You want to create a lightweight Web-based data entry and reporting solution without a lot of additional effort.
- You want to implement some degree of data security and data backup and restore, and you're willing to make a small time investment to do so.

Use SQL Server when

- Your data exceeds Access's operating limits.
- You have a large amount of multidimensional data.
- You have a large number of people that need simultaneous access to the data.
- You need to synchronize a large amount of data over a large number of remote locations.
- You need to perform a large number of transactions that can be rolled back as a group if certain conditions aren't met.
- You have advanced or unique needs for data storage; data backup, retrieval, and restore; and data logging and audit tracking.
- You have advanced or unique solution development needs such as data entry triggers and stored procedures.
- You want to import, export, or synchronize data across a wide array of database management systems.
- You want to create an end-to-end, Web-based solution to interact with and manage your data.
- You want to take advantage of more powerful computing resources for faster data access and data management.

For more information on Access and SQL Server, see the following:

- *Microsoft Office Access 2007*: `http://office.microsoft.com/en-us/access/FX100487571033.aspx`
- *Access 2003 Product Information*: `http://www.microsoft.com/office/access/prodinfo`
- *SQL Server 2005 Overview*: `http://www.microsoft.com/sql/prodinfo/overview`