IN THIS CHAPTER

- Undertanding what SQL does
- Getting clear on what SQL doesn't do
- Weighing your SQL implementation options

Chapter 1
Getting to Know SQL

In the early days of the relational database management system (RDBMS), there was no standard language for performing relational operations on data. A number of companies came out with RDBMS products, and each had its own associated language. However, differences in syntax and functionality made it impossible for a person using the language of one RDBMS to operate on data that had been stored by another. The creation of SQL solved this problem, but SQL is a continually evolving language that changes with each official release (the most recent being in 2023). This chapter explores just what SQL is (and isn't). It also takes a look at using SQL in some different database packages.

Knowing What SQL Does

SQL (pronounced *ess cue el*) is a software tool designed to deal with relational database data. It does far more than just execute queries. Yes, you can use it to retrieve the data you want from a database using a query. But you can also use SQL to create and destroy databases, as well as modify their structure. In addition, you can add, modify, and delete data with SQL. Even with all that capability, SQL is still considered only a *data sublanguage,* which means that it doesn't have all the features of general-purpose programming languages such as C, C++, C#, or Java.

COPYRIGHTED MATERIAL

SQL is specifically designed for dealing with relational databases, so it doesn't include a number of features needed for creating useful application programs. As a result, to create a complete application — one that handles queries, as well as provides access to a database — you have to write the code in one of the general-purpose languages and embed SQL statements within the program whenever it communicates with the database.

Knowing What SQL Does Not Do

Before we can tell you what SQL doesn't do, we need to give you some background information. In the 1930s, computer scientist and mathematician Alan Turing defined a very simple machine that could perform any computation that could be performed by any computer imaginable, regardless of how big and complex. This simple machine has come to be known as a *universal Turing machine.* Any computer that can be shown to be equivalent to a universal Turing machine is said to be *Turing-complete.* All modern computers are Turing-complete. Similarly, a computer language capable of expressing any possible computation is said to be Turing-complete. Practically all popular languages, including C, C#, C++, BASIC, FORTRAN, COBOL, Pascal, Java, and many others, are Turing-complete. SQL, however, is not.

Because standard SQL is not Turing-complete, you can't write an SQL program to perform a complex series of steps the way you can with a language such as C or Java. On the other hand, languages such as C and Java don't have the data-manipulation capabilities that SQL has, so you can't write a program with them that will efficiently operate on database data. There are several ways to solve this dilemma:

- Combine the two types of language by embedding SQL statements within a program written in a host language such as C.
- Have the C program make calls to SQL modules to perform data-manipulation functions.
- Create a new language that includes SQL, but also incorporates those structures that would make the language Turing-complete. (This is essentially what Microsoft and Oracle have done with their versions of SQL.)

All three of these solutions are offered by various vendors.

Choosing and Using an Available RDBMS Implementation

SQL by itself isn't all that useful — you need a platform that stores the data itself and uses SQL to create, read, update, and delete (often called CRUD) the data. This is where the RDBMS comes in.

The RDBMS is a program that stores data in a manner that makes it easy to retrieve the data as quickly as possible. Storing data in a typical file isn't efficient, because in order to find a specific data item, the program would have to read through the entire file until it got to that data.

An RDBMS system uses various methods to store and index data so it can quickly find a specific data record, based on the SQL statement it's processing. There are plenty of different RDBMS programs available these days, each with different features to help increase data retrieval performance. In the following sections, we fill you in on some of the more common RDBMS programs available today.

Microsoft Access

Microsoft Access is an entry-level RDBMS with which developers can build relatively small and simple databases and database applications. It's designed for use by people with little or no training in database theory. You can build databases and database applications using Access, without ever seeing SQL. However, you can opt to use SQL in Access if you so choose.

Access runs under any of the Microsoft Windows operating systems, as well as Apple's macOS, but not under Linux or any other non-Microsoft operating system.

To reach the SQL editor in Access, do the following:

1. **Open a database that already has tables and at least one query defined.**

   A great place to start is with the Northwind Traders Starter Edition database provided as a free download with Access. The database includes a built-in mini-application that uses Access forms to help query and insert data. After you

download the database, the application portion automatically starts, showing a form that asks you to create a user account for the application. After you log into the application, you see a database window that looks something like Figure 1-1, with the default Home tab visible. The icon at the left end of the Ribbon is the icon for Layout View, one of several available views. In this example, the pane on the left side of the window shows the different tables, forms, reports, queries, and scripts that have been created as part of the Northwind Traders Starter Edition database.

2. **Click the Queries entry in the pane on the left, and then double-click the qryProductOrders query.**

   The default view shows the data that's a result of the query.

3. **To see how the query is constructed, click the View icon at the top, and then select Design View.**

   The Design View for the query is shown in Figure 1-2. At the top is a graphical representation of the tables involved in the query, and below that is a list of the data fields that are retrieved in the query.

4. **Choose SQL View from the View drop-down menu.**

   Doing so shows the view displayed in Figure 1-3. It's the SQL code generated in order to display the result of the Team Membership of Paper Authors query.

   As you can see, it took a pretty complicated SQL statement to perform that Product Order query.

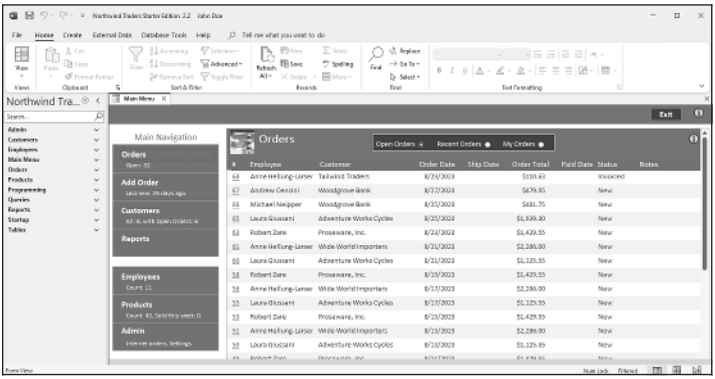

FIGURE 1-1: A Microsoft Access 365 database window running the Northwind Traders Starter Edition database.

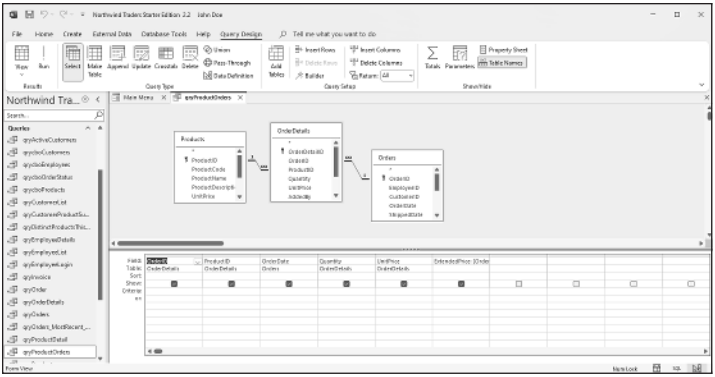

FIGURE 1-2: The Design view of the qryProductOrders query.

FIGURE 1-3: The SQL view of the qryProductOrders query.

When you're a true SQL master, you may want to enter a query directly using SQL, instead of going through the extra stage of using Access's QBE facility. When you get to the SQL Editor, which is where you are right now, you can do just that. Step 8 shows you how.

5. **Delete the SQL code currently in the SQL Editor pane and replace it with the query you want to execute.**

   For example, suppose you wanted to display all the rows and columns of the PRODUCTS table. The following SQL statement will do the trick:

```
SELECT * FROM PRODUCTS ;
```

6. **Execute the SQL statement that you just entered by clicking on the big red exclamation point in the ribbon that says Run.**

   Doing so produces the result shown in Figure 1-4, back in Datasheet View. This is a listing of all the data records stored in the PRODUCTS table.

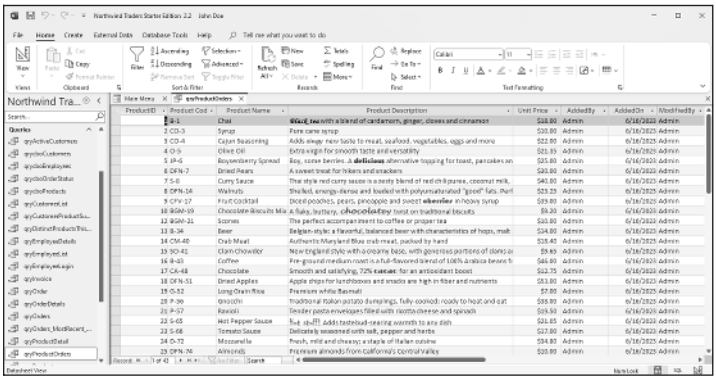

FIGURE 1-4: The results of the query to display all the data in the PRODUCTS table.

WARNING

Don't save your new query because it will replace the standard qryProductOrders query in the Northwind Traders Starter Edition database. Just exit out without saving your changes.

Microsoft SQL Server

Microsoft SQL Server is Microsoft's entry into the enterprise database market. It runs only under one of the various Microsoft Windows operating systems. The latest version is SQL Server 2022. Unlike Microsoft Access, SQL Server requires a high level of expertise in order to use it at all. Users interact with SQL Server using Transact-SQL, also known as T-SQL. Additional functionality provides the developer with usability and performance advantages that Microsoft hopes will make SQL Server more attractive than its competitors. There is a free version of SQL Server 2022, called SQL Server 2022 Express Edition, that you may think of as SQL Server on training wheels. It's fully functional, but the size of database it can operate on is limited.

IBM DB2

IBM DB2 is a flexible product that runs on Windows and Linux PCs on the low end all the way up to IBM's largest mainframes. As you would expect for a DBMS that runs on big iron, it's a full-featured product. As with Microsoft's SQL Server, to use DB2 effectively, a developer must have received extensive training and considerable hands-on experience.

Oracle Database

Oracle Database is another DBMS that runs on PCs running the Windows, Linux, or macOS operating system, and also on very large, powerful computers.

SQL Developer is a free graphical tool that developers can use to enter and debug Oracle SQL code.

A free version of Oracle, called Oracle Database Express Edition, is available for download from the Oracle website (`www.oracle.com/database/technologies/appdev/xe.html`). It provides a convenient environment for learning Oracle. Migration to the full Oracle Database 11g product is smooth and easy when you're ready to move into production mode. The enterprise-class edition of Oracle hosts some of the largest databases in use today. (The same can be said for DB2 and SQL Server.)

Sybase SQL Anywhere

Sybase SQL Anywhere is a high-capacity, high-performance DBMS compatible with databases originally built with Microsoft SQL Server, IBM DB2, Oracle, and MySQL, as well as a wide variety of popular application-development languages. It features a self-tuning query optimizer and dynamic cache sizing.

REMEMBER

Tuning queries can make a big difference in their execution time. *Tuning* a query means making adjustments to it to make it run faster. *Dynamic cache sizing* means changing the size of the cache memory available to a query, based on the resources that the query needs to run as fast as possible. We talk about query tuning in Chapter 4.

MySQL

MySQL is the most widely used open-source DBMS. The defining feature of open-source software is that it's freely available to anyone. After downloading it, you can modify it to meet your needs and even redistribute it, as long as you give attribution to its source.

One amazing feature of MySQL is that it offers multiple ways of storing and managing data, which are called *storage engines.* The most feature-rich of these is the InnoDB storage engine, which provides many of the advanced database features found in commercial databases such as Microsoft SQL Server.

Another popular storage engine is the MyISAM storage engine, which is particularly noted for its speed. Although it lacks many of the advanced features found in the InnoDB storage engine, the MyISAM storage engine is amazingly fast with simple data queries, making it a popular choice for web-based applications. The MySQL server runs under Windows and Linux, but not under IBM's proprietary mainframe operating systems. MySQL is supported by a large and dedicated user community, which you can learn about at `www.mysql.com`.

PostgreSQL

PostgreSQL (pronounced *POST gress CUE el)* is another open-source DBMS, and it's generally considered to be more robust than MySQL, and more capable of supporting large enterprise-wide applications. It's also supported by an active user community. PostgreSQL runs under Linux, UNIX, Windows, and IBM's z/OS mainframe operating system.