Microsoft Access for Beginners

Second Edition

An introduction to Microsoft Access and relational database design

Andrew Comeau

Microsoft Access for Beginners

Andrew Comeau
P.O. Box 770253
Ocala, Florida 34477
acomeau@drewslair.com
http://www.drewslair.com

Contents

This is a free preview of the book “Microsoft Access for Beginners”. The full book contains seven additional chapters and appendices. See the end of this preview for links to the sites where you can download free samples in the format of your choice.

Introduction

This book started out as a series of articles on my website at Drewslair.com. I decided to write the series after answering questions about Access as a result of my work with it and talking to people who didn't understand exactly what Access was or how it compared to other programs in Microsoft Office. I've seen many examples of databases created by people who approach Access in the same way as they would use Excel, setting up one table to hold everything and duplicating data as necessary.

It's understandable that many people try to use Access this way as more people are familiar with the spreadsheet concept than database concepts. Access presents tables that look something like the spreadsheet format that people are used to and, since it's offered as part of the Microsoft Office suite, it's easy to see it in relation to the other programs and overlook its power as a standalone product.

Microsoft Access is significantly different than any of the other Office applications, however. While programs like Word and Excel focus on documents, Access files are database applications that contain a variety of objects such as tables, forms and reports, all of which work together to organize and manage your data. The most sophisticated Access databases function as programs in and of themselves with user interfaces made up of forms, reports and custom menu systems. Access is even used by some programmers to design sophisticated, standalone systems for fields such as document management, customer relations and inventory tracking.

Before you design the next killer app however, you need to learn some basic database and design concepts so you can use Access to its full potential. I consider this book to be a mid-level introduction to Microsoft Access as I don't spend a lot of time on step-by-step instructions. Instead, I introduce you to the basics of the

program and provide enough information so you will understand how to correctly create tables and design queries to read from them. Then I show you what you can do with forms and reports. I also provide an introduction to SQL so you will better understand what queries do. I don't provide exhaustive detail on every feature but I do show you how to use Access *well*.

Most books about a program like Access focus only on the latest version but I take a different approach. As of this writing, the current version of Access is Office Access 2010. I primarily focus on the 2007 and 2010 versions but I also realize that there are still people using earlier versions so I've tried to ensure that these users will find the book helpful as well. Most of the concepts I write about here are common to all versions of the program and, where necessary, I give specific instructions by version.

A Word on Design Wizards

For many of the objects in Access such as tables and forms, there are wizards that you can use to quickly create an object by answering a few questions and making a few selections. Microsoft Access even enables quick, wizard-based design of entire applications. The result is a ready-made solution designed to do exactly what you specified, whether that's what you wanted or not. In this book, I rarely mention the wizards because I don't use them for anything significant when designing a database. I don't believe that they are good tools for creating professional applications.

If you want to 'create' something you know nothing about and spend a lot of time thinking that it doesn't meet your needs, then use the wizards. You don't need my help on that. On the other hand, if you want to take ownership of the project and be able to support it, do it yourself. You'll develop your skills that way and it's a lot more fun. Eventually, you will have a library of your own work that you can consult

whenever you want to figure out how to do something. The JobSearch 2010 demo is an example of what can be done without design wizards.

Throughout this book, I will show you how to create things on your own so you will learn how to design programs with Microsoft Access.

I welcome your comments and questions concerning the material presented here. You can e-mail me with any comments at acomeau@drewslair.com.

Enjoy!
Andrew Comeau
December 2012

Using the Sample Database

Throughout this book, you'll see references and screenshots from a sample application that I adapted for use here. JobSearch 2010 (now known as Job Search Plus) is a program that I designed to help individuals organize all the information related to a job search including job lead details, company and contact information and search activities. It also features flexible reporting and an interface that demonstrates how Access can be used to design a professional application. The full program is available as a free download from Drewslair.com with multiple versions in order to accommodate people running different versions of Microsoft Access. There's even a version that includes the free Access runtime for those who do not have an Access installation.

For this book, I've created a special version of JobSearch 2010 that allows access to the design environment and the code used to enable many of the features in the program. This is a stripped-down version of the program that omits features for which I did not want to release the code. You can find more information, an online manual and download links to all versions of the program at http://www.andrewcomeau.com/pages/jsmanual.htm.

To install the program, simply unzip the file to a chosen location on your computer and double-click on the MDB file to start it with your installation of Access. The database requires Access 2002 or later and will also work with the Access 2007 run-time which is available as a free download from Microsoft.

Macro Security and Trusted Locations

Starting with Access 2003, Microsoft took extra steps to guard against the possibility of malicious code being distributed as part of Access files. Access now notifies the

user when opening a file of the possibility of malicious code. Depending on your version and settings, Access will even deactivate this code by default. As the sample database depends on VBA code for its operation, you need to be aware of the steps to allow the code to run.

Access 2003

Access 2003 had some relatively simple precautions against potentially unsafe code. Assuming that all the service pack updates are installed, you might just see a message like this when opening the sample database.

Figure S.1 - Access 2003 unsafe code warning

Clicking on the Open button will open the database and run the necessary code. You can turn off this warning if you like by changing the macro security options under the Tools -- > Macro --> Security menu. In this example, I have the security set to Low which will run the code without warnings. The options under the Trusted Publishers tab enables the user to trust content from certain publishers although this requires the content to be signed with a security certificate which you might never encounter with Access databases.

Access 2007 and 2010

In Access 2007, Microsoft introduced the Trust Center which enabled the user to trust content in specific locations on the computer.

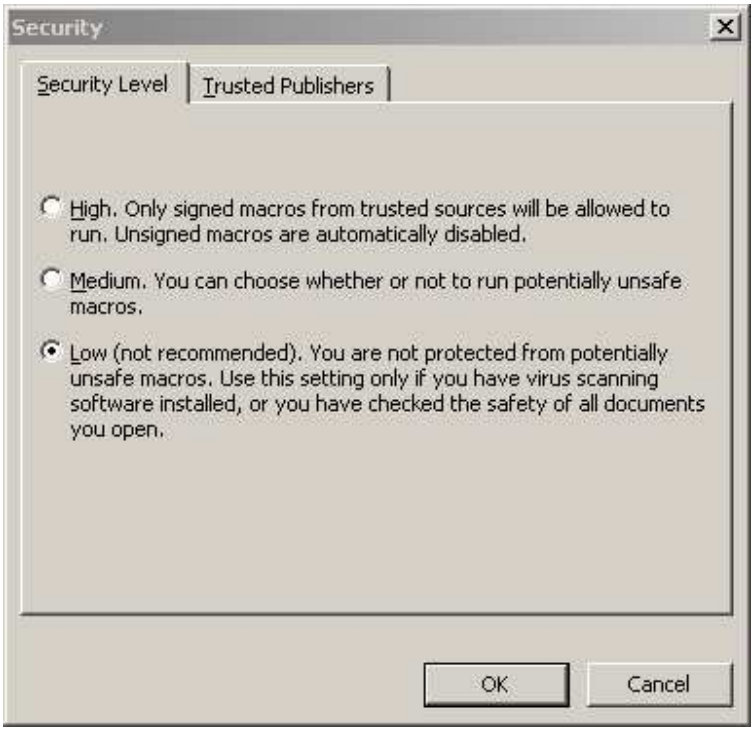

Figure S.2 - Macro security options

This lets you define places on your computer where you know that the content is safe rather than changing the security setting for the entire Access application. When you open the sample database in Access 2007, you might see the following message bar:

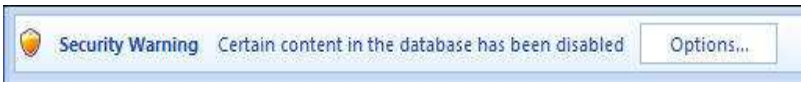

Figure S.3 - Access 2007 active content security warning

You can enable the content for the current session by clicking the Options button and selecting 'Enable this Content'. At the bottom of the Options is a link to open the Trust Center which can also be opened from the Access program options screen

available from the Office 2007 Button. The Trust Center dialog provides options to manage the list of trusted locations on the local system. Adding locations such as the Windows Desktop or My Documents folder will allow content in these locations to be run without restriction.

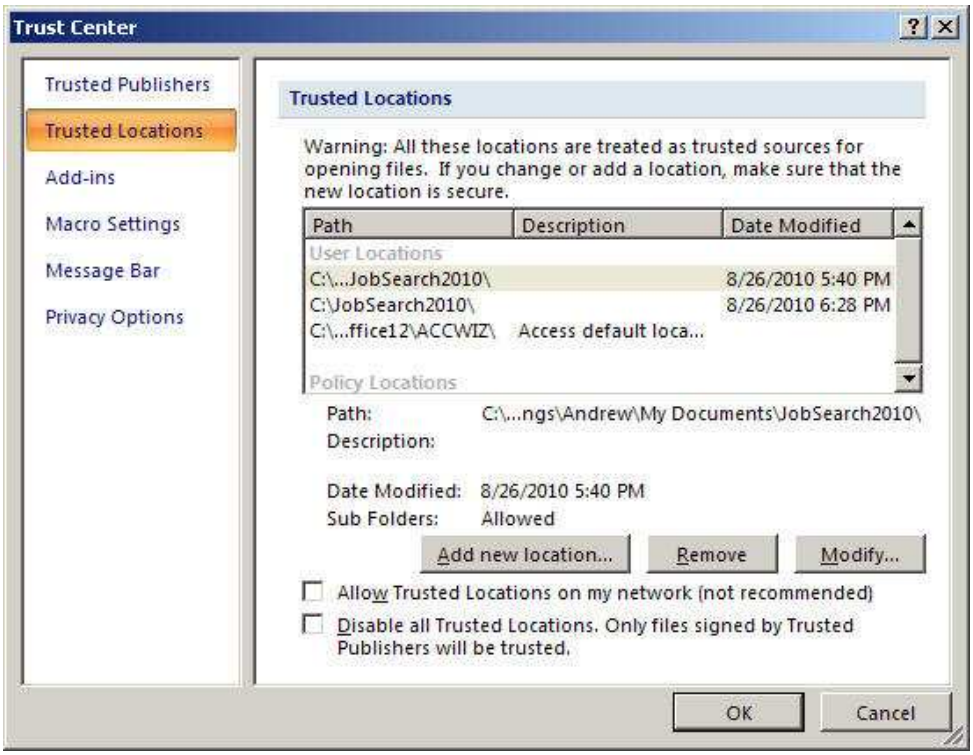

Figure S.4 - Access 2007 Trust Center

For more information, see the MSDN article "Security Considerations and Guidance for Access 2007", the link for which is available in the Suggested References list at the end of this book. You can also find the article by searching on the above title at msdn.microsoft.com.

Access 2010

Access 2010 takes the trust settings down to the document level. Clicking the 'Enable Content' button on the security warning will permanently enable that

database at that location. If you copy the database to another location or rename the database, you will get the warning again. The Trust Center is still available from the Actions Options panel which is available on the Access 2010 File tab.

The Trust Center in Access 2010 also has more options including the option to disable document trusting and clear the list of previously trusted documents.

Interface Guide

As mentioned in the Introduction, this book accommodates users of multiple versions of Microsoft Access. With Access 2007, Microsoft introduced some major changes to the Office interface that required users to learn where everything was all over again. These changes were refined in Access 2010. I've included a short explanation here so you'll know what I'm referring to in upcoming chapters.

In Access 2003 and earlier, Access used a standard menu and toolbar interface. Figure IG.1 shows the demonstration program with a customized menu system in Access 2002.

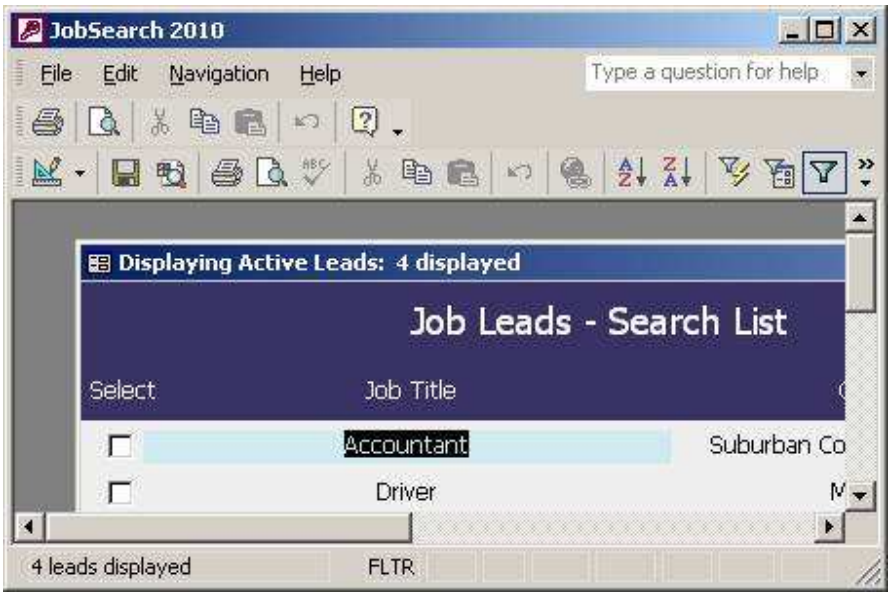

Figure IG.1 - Access 2002 interface

Office 2007 introduced the Office Button and Ribbon. The Office Button is a pull-down menu that replaces the old File menu. You can use it for file and print management functions, to close the database and to access the program options.

The Ribbon groups all of the program functions under tabs according to category. Figure IG.2 shows a customized ribbon that I created for the full version of JobSearch 2010.

Figure IG.2 - Access 2007 interface

Office 2010 did away with the Office Button and returned its functions to a File tab on the Office Ribbon. Customizing the Ribbon is also easier in 2010. In Access 2007, it was necessary to create a special table containing custom XML code, Office Access 2010 provides an interface for adding and removing controls from the ribbon.

While the Ribbon interface took some time to get used to, I feel that it was a good move for Office. With the increasing number of features available in the programs, the ribbon makes it easier for users to find the one they need. The return of the Office Button features into the main ribbon interface was also an improvement.

If you're generally familiar with the way things work in Windows and Office, then you shouldn't have much of a problem switching between versions of Access but I do try to make things as easy as possible for all users. While many features maintain the

same interface between the various versions, I will provide an explanation for how to accomplish something in each version as necessary.

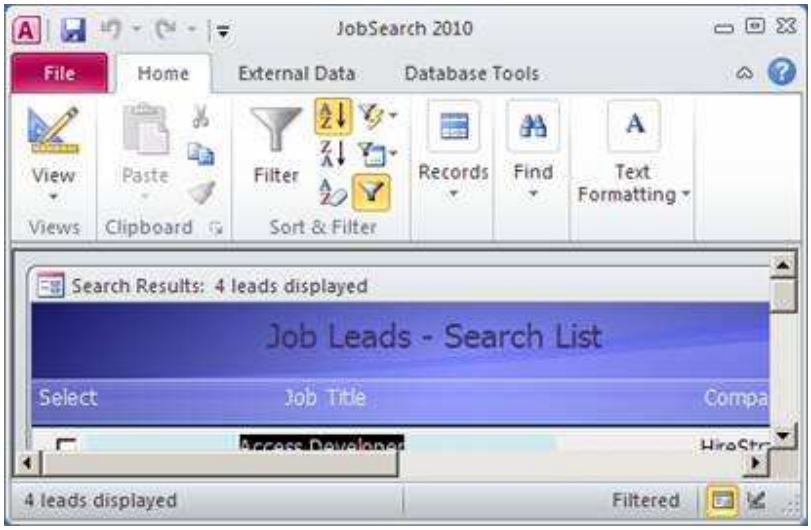

Figure IG.3 - Access 2010 interface

Chapter I - Creating the Database

A Different Set of Tools

For many people, Microsoft Access is probably the most enigmatic of all the Microsoft Office applications. It's only included in the Professional editions of the suite or as a standalone package which means that most home users won't have it installed. People will often hang out in Word, Excel and Outlook but never have reason to wander into the database portion of the suite. The interface itself is unfamiliar to the average user who is more accustomed to a spreadsheet or blank word processing document than the collection of objects that Access presents. Even in Access 2007 and later, the ability to create entire applications from templates departs from the document focus that many users expect.

Those who do explore the features in Microsoft Access, however, find that it is likely the most powerful of all the Office applications with its ability to store and organize huge amounts of information in related tables, then query and sort the information as needed and finally present the data in attractive, professional reports. Custom data entry forms make it easy to enter and review data and guard against data entry errors. All of this is available even before a user might start playing around with Visual Basic for Applications, the programming language that enables users to create virtually any type of interactive application and turns Microsoft Access into a full development environment.

What is Access?

As a Windows desktop database, Microsoft Access provides many of the features of a *Relational Database Management System (RDBMS)*. In order to understand what that means, we have to take the term a bit at a time.

A *database* is any collection of information that can be organized by a standard set of fields. Your address book is one classic example. Each entry in the book represents a record with fields such as Name, Address, Phone Number, etc.. A dictionary could also be considered a database with its standard entries including the word, pronunciation and definition. Databases might include relatively few records or tens of thousands of records, depending on the need.

A database *management system* such as Access enables this data to be stored electronically according to a set of rules. The tables in Microsoft Access organize the data and provide features that are common to databases such as indexes to make it easier to search and sort the data, primary keys that identify each record with a unique value and validation rules that help to ensure the data conforms to the necessary limitations. Access also provides a way to retrieve and manipulate the data through queries. Finally, a database management system enables the creation of new databases as needed and can be used to manage large collections of separate database applications.

Access is also a *relational* database system. There are different types of database systems that are structured differently in order to serve various purposes. In a relational database:

1. Data is organized into tables which are sometimes called *relations* by database experts. This is because each table has a subject (i.e. Companies) and every field in a given table is supposed to be directly related to the subject of the table.
2. Tables can be linked or related to each other. This is demonstrated in the sample database by a link between the Leads and Activities tables. Since there can be more than one activity for each job lead, the Activities details are placed in a separate table and the two tables are linked by an ID number assigned to the job lead. This eliminates the duplicate entry of information

such as job title or description. This, in turn, reduces the size of the database and the possibility of data entry error and database corruption.

This is how Access differs from a program like Excel. While Excel organizes information and allows for calculations and analyses, Access takes it a step further and enables relationships between different categories of data and faster analysis of information.

Database experts will point out that Access lacks a couple of the features that would fully qualify it as a relational database system, however it is still a good tool for small to medium sized database applications and provides an excellent training ground for database and programming concepts.

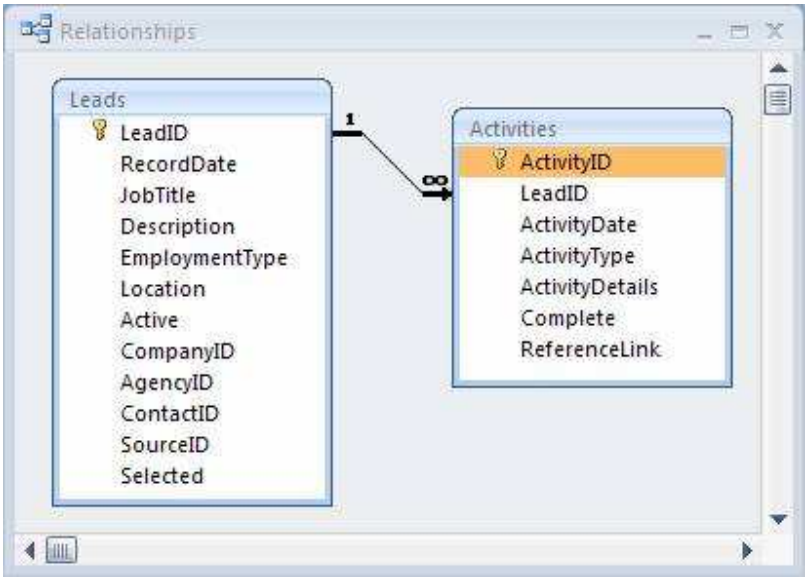

Figure 1.1 - Relational databases enable you to link different categories of data.

File Formats

Microsoft Access uses a single file to store all of the objects used in a database application including tables, queries and reports. This differs from other desktop

database systems which might store queries and other items in separate files. The use of one file makes the databases very portable.

Despite the use of one file to store everything, Access applications can make use of external files by linking to data in other Access databases, Excel workbooks and a variety of other formats including text and HTML files. This ability is commonly used to separate the data tables from the interface portion of an Access application by storing the tables in one Access database file (referred to as the back end) and linking to them from another Access file which stores the forms, reports and other items (the front end). Access can also link to data in other file formats, including Excel and Microsoft SQL Server. Access references the links to external data sources just like any other table. This has a number of advantages, such as enabling multiple users to access the data at the same time from different locations and making it easier to change reports and forms without disrupting access to the data.

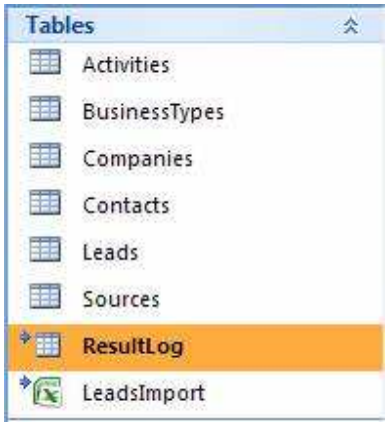

Figure 1.2 - Access is able to link to other data formats.

Figure 1.2 shows the table listing from an Access 2007 database that includes a couple of linked sources.. The ResultLog item is a linked Access table while LeadsImport is a link to an Excel file. When linking to other formats like Excel, Access will look for data structures such as worksheets or tables that can be read

like an Access table and provides a wizard that will guide you through the process of linking to the data.

Microsoft Access has many file extensions that can be used for different types of applications such as templates and add-ins but there are only a couple that you need to be immediately familiar with.

In Access 2007 / 2010, the ACCDB file extension indicates a typical database that you would create and work with. The ACCDE file extension indicates an Access database that has been protected against changes to the objects within it. The data can still be changed but no design changes can be made to the tables or other objects. The corresponding file extensions in Access 2003 and earlier were MDB and MDE. As with other Office applications, Access is able to open and save files created in other versions of the software although there are limitations in the creation of the protected databases (MDE / ACCDE).

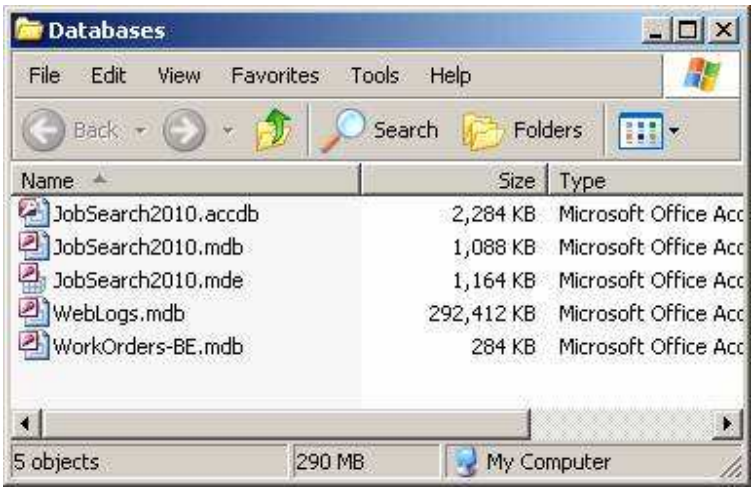

Figure 1.3 - A few of the extensions and file formats used by Access.

Protected Access databases are also used to protect programming code and design details within the database from being viewed or modified. The demonstration database for this book is distributed as an unprotected MDB file created in Access

2002. It can also be opened and modified in any version of Access after that. This database is a stripped-down version of JobSearch 2010 which I designed in Access 2007 and distributed as a *protected* Access database in both the 2002 and 2007 formats. The unprotected version designed for this book omits a few features and a generous amount of code that I did not want to make public.

Creating a New Database

Once installed, Microsoft Access adds itself to the New Object menu in Windows. From the File menu in any folder, or by right-clicking on the desktop, select New and then 'Microsoft Access Application'. This will create a blank database which you can then rename and open within Access. The other way is to use the New command from the Office button in Access 2007 or from the File menu in all other versions and then choose Blank Database.

Access offers a number of pre-designed templates and wizards that can help you create entire applications just by answering a few questions. These are great examples of applications and worth studying if you're new to Access but, as I said in the Introduction, I would not recommend them for use with real data. As complex as Access applications and the requirements for them can be, it is in your best interests to learn what you need to know to design and support your own database.

When you first open a database in Access, you can press F11 to show the database window if it's not already visible. This window will display all of the objects in the database.

Figure 1.4 shows the Access 2007 window with the menu that enables you to change the objects listed. In Figure 1.5, you can see an example of the window from the sample database.

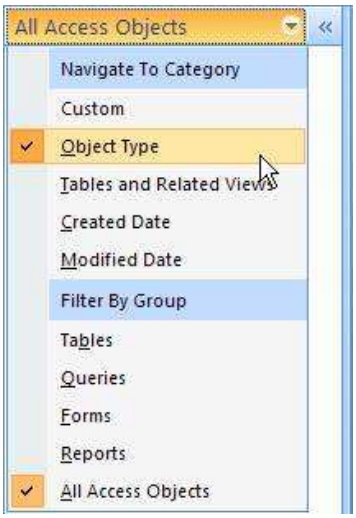

Figure 1.4 - Access 2007 database window settings

Database Objects

Other Office applications deal primarily with documents such as letters and spreadsheets but Access takes a different approach; the database file can contain many objects of different types which work together to provide the functions you need in your database program.

Tables are the heart of the database, storing your data and organizing it for quick retrieval. Access data is, ideally, stored according to certain principles that organize the data by subject and eliminate duplication. I will provide more details on these principles in Chapter II. In Chapter III, you'll see how to design the tables and use table properties in order to get the maximum benefit out of Access as a database.

Queries provide a way to retrieve and manipulate data stored in the tables. Access uses Structured Query Language (SQL) to perform any type of operation that's needed on the data or even the structure of the tables themselves. Beyond reading and manipulating data, queries can act as data sources similar to tables, performing intermediary processing of the data and passing results on to other queries for

further processing. Chapter IV provides additional information on queries and a beginners' guide to SQL is presented in Chapter V.

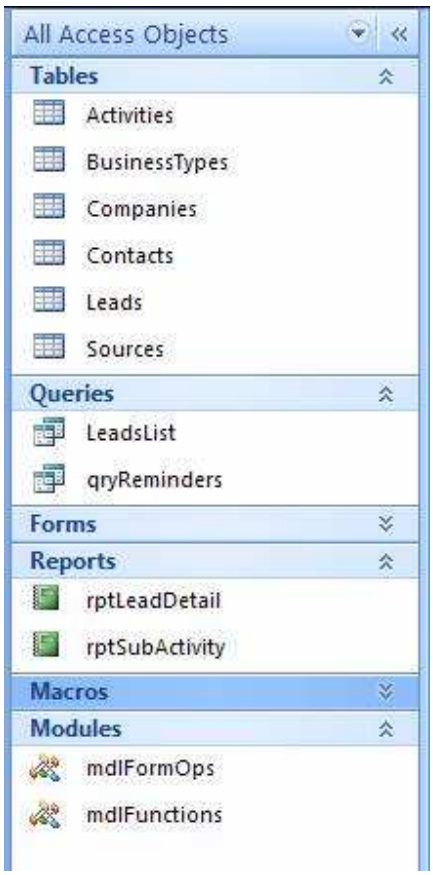

Figure 1.5 - The database window shows all of the objects in the database in one place.

Forms provide a fast and customizable way to manage data within your program. Entering data directly into tables can be slow and prone to errors. Forms offer a wealth of tools and controls that will speed up your data entry and help prevent user errors. Chapter VI details some of the possibilities that are available with forms and how they can be used to create a professional user interface.

Chapter VII will get you started with reports which are essential to any data management system. The report designer is one of the greatest strengths of

Microsoft Access. Access is loaded with features to help you analyze your data and all of these features come together in an easy-to-use report designer that will help you communicate the data to others within your organization.

Learning to use macros and modules is the next step once you've learned the basics of Microsoft Access. Chapter VIII introduces macros which are the most basic of programming features in Access, enabling you to automate basic tasks by scripting sequences of steps for Access to follow. Modules are used to hold advanced custom functions that you can create using Visual Basic for Applications (VBA). You'll see more information on them in Chapter IX.

Compact and Repair

Microsoft Access database files are designed to hold up to two gigabytes of information, including your tables, forms, reports and other objects. The problem is that once the file grows to accommodate new data or design changes, it doesn't automatically let go of that space when data or objects are removed from the database. This can cause the database to grow quite large, especially during the design process when a lot of objects are being changed.

The solution for this is to use the Compact and Repair function in Access.

- Prior to Access 2007, the Compact / Repair function is available from the Tools --> Database Utilities menu.
- In Access 2007 and later, you can find it by clicking on the Office button and selecting Compact and Repair from the Manage menu.
- In Access 2010, the Compact and Repair button is moved to the Database Tools tab of the Access ribbon.

Access will exit the database and create an entirely new database file from the old one, releasing the extra space. This process will also attempt to repair certain types of corruption in the database file.

Compact and Repair also resets certain counters in the database. For example, in a future chapter, you'll learn about AutoNumber fields which automatically generate a new unique number for each record that's entered into a table. These are often used for ID fields and are usually sequential. Compact and Repair will reset this number so that the next number used is continuous from the last one. This can be useful when you are first designing a database and entering and deleting a lot of test records in tables. Before releasing the database to other users, you should run a Compact and Repair which will reset those AutoNumber counters.

Compact and Repair is not foolproof for eliminating all wasted space or fixing errors. The Access file format holds on to temporary data that is not released even when compacting and there are some problems it won't fix. This is one reason backups are important. Be sure to backup your database often and before making any major changes to the design.

Protecting your Database Design

Earlier, I mentioned the MDE and ACCDE formats sometimes used by Microsoft Access. These are protected database formats in which all design features are disabled and all editable code has been removed. Sometimes, if you are distributing your database application to other people, you might want to use one of these formats to prevent unauthorized users from making design changes or prevent access to your code.

Creating a protected database is pretty simple in any version of Access.

- In Access 2010, select “Save & Publish” from the File menu and select “Make ACCDE” under the advanced options.
- In Access 2007, select “Make ACCDE” under the Database Tools tab of the Office Ribbon.
- In Access 2003 and previous, select the “Make MDE File” menu option under the Tools --> Database Utilities menu.

The Microsoft Access file that you are trying to convert must be saved in the same version of Access in which you are working before creating a protected file, i.e. if you are working in Access 2007, you must be working with an ACCDB file in order to make an ACCDE. You cannot make an ACCDE file from an MDB. If the Make ACCDE / MDE command is grayed out, it’s probably because the file you’re working with was saved under a different version of Access.

When you create an ACCDE or MDE, a new file will be created separately from your ACCDB or MDB file. It’s important to keep the original file in which you can make design changes because the protected database that you just created cannot be used to recover the original file.

Conclusion

Microsoft Access is a little bit of a paradox; marketed as part of the Microsoft Office suite and yet providing enough functionality and power to stand on its own. Its ability to analyze data from a host of other programs makes it a valuable part of the Office environment and accessible to the average user. Yet, there are also people who have started programming careers by using Access to master the database and programming concepts involved in creating a great database application. Whatever your goal with Access, the following chapters will provide you with all you need to know to start building databases and add some very valuable skills to your resume.

Chapter II - Organizing the Data

A Structure for Your Data

In Chapter I, I explained how a relational database enables you to categorize data by subject within your database and then create relationships between different categories so the data can be viewed according to those relationships. In this chapter, I want to explain the system that's used to do this. Later, I'll go into more detail about creating a table and using the available settings. My interest here is to show you the proper way to design what's called the database *schema* (pronounced SKEE-mah) which is the collection of related tables and other objects that form the structure of the database.

Beginning in 1970, a computer scientist at IBM, Edgar F. Codd, developed a set of rules by which data could be organized in relational databases such as Microsoft Access (although neither Microsoft *or* Access would be around for a few years at that point). This system is called *Database Normalization* and has become the accepted standard for designing relational databases. The purpose of normalization is to simplify the updating of data and avoid inconsistencies and duplication within the database. It also makes adding new types of data to the database structure easier because there is already a context to build on.

While there are several rules in database normalization, also referred to as *Normal Forms*, there are only three that you really need to be concerned with when working with Microsoft Access. If you implement these three in your database, you will have a very stable and well-organized system.

The basic rules of normalization that you need to know at this point can be summed up as follows:

1. Table fields should be designed to hold single values that represent basic units of information.

2. There should be no repeating fields or groups of fields in a table. (i.e. Item1, Item2, Item3, etc..).
3. All fields in any given table should relate directly to that table's entire primary key and non-key fields should not be dependent on each other.

This is a very basic summary and I will present a more detailed explanation later in the chapter. First, let's see the rules in action in the demonstration database.

Tables and Relationships

After downloading the demonstration database, start by opening the database in Access. Press the F11 key to show the database objects window. The database window will show you all the tables and other objects in the database.

Figure 2.1 - In all versions of Access, the database window is accessed by pressing F11.

Although F11 works in all versions, Access 2007 introduced some radical changes to the Access interface. While many screens look the same, some are different enough to be worth mentioning and I will note the differences as necessary.

One of the goals of a user-friendly Access database is to minimize the amount of work required to enter and retrieve the information, thereby reducing workload and the possibility of error. This is where the ability to group data into separate tables comes in handy. These tables organize the data so that it's easier to find and, as you'll see, help to eliminate the need to enter the same information over and over

again. The tables you create and the relationships between them are the backbone of your database. Once you have the tables designed correctly, you're halfway there.

The Leads table is the central table in the database and it contains the essential information for each job opportunity. In accordance with database normalization, Leads only contains the information that specifically relates to the job lead such as the date recorded, job title and description.

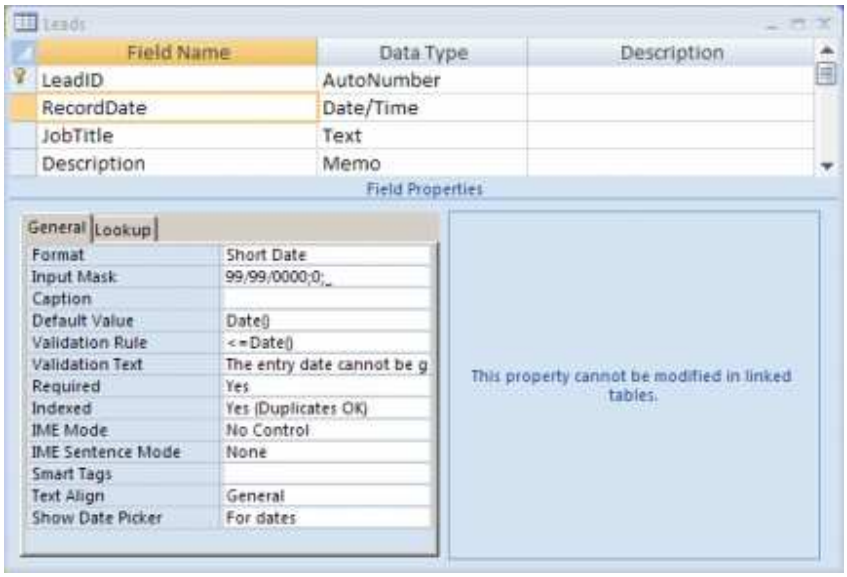

Figure 2.2 - The table design view provides complete access to settings for the table and the individual fields.

If you view the design of the Leads table by right-clicking on the table in the database window and selecting Design View, you will see by the key symbol next to the field name that the LeadID field is identified as the table's Primary Key. This means that the field contains a value that is unique for each record in the table, enabling the database to identify that particular record. Access encourages you to create a primary key for every table that you design.

In the second column of the design view, you'll see the data types assigned to each field. Assigning specific data types to fields enables the database to more efficiently store the data and enforce rules for entering the data as needed. A number of types

are shown in the Leads table including Date/Time which stores dates and enables date based calculations, the Yes/No type which stores a simple Yes or No value appropriate to checkbox options and the Number type which allows for a wide range of numeric formats. The data type assigned to the LeadID field is called AutoNumber. This is a unique number assigned by the database to every record that the user enters. This value cannot be edited by the user.

For example, if you were creating an employee database, you would not want to use this number as an employee number within the company records because there is no guarantee that it won't skip values if records are cancelled as they're being entered. It will also assign the numbers in the order that the records are entered while employee numbers are often assigned in order of hire. The main, and often the only, purpose of an AutoNumber primary key field such as LeadID is to create an identifier that the database can use to refer to each record.

The Activities table in the sample database, which lists actions for each job lead such as applications, correspondence and follow-ups, also has a field titled LeadID but the data type is Number instead of AutoNumber. In terms of the relationships between Activities and Leads, the LeadID field in Activities is referred to as the *Foreign Key* while it's called the *Primary Key* in Leads. This is because the value originates in Leads as an AutoNumber that identifies a specific job lead and each record in the Activities table that relates to that job lead will use the identifying number from Leads to refer to the correct Leads record. In Figure 2.3, you can see a relationship drawn between the two tables. This relationship enforces the link between the LeadID field in each table. In a later chapter, you'll see how forms use this relationship to automatically insert the correct LeadID value in the Activities table to link the records there to the right records in the Leads table.

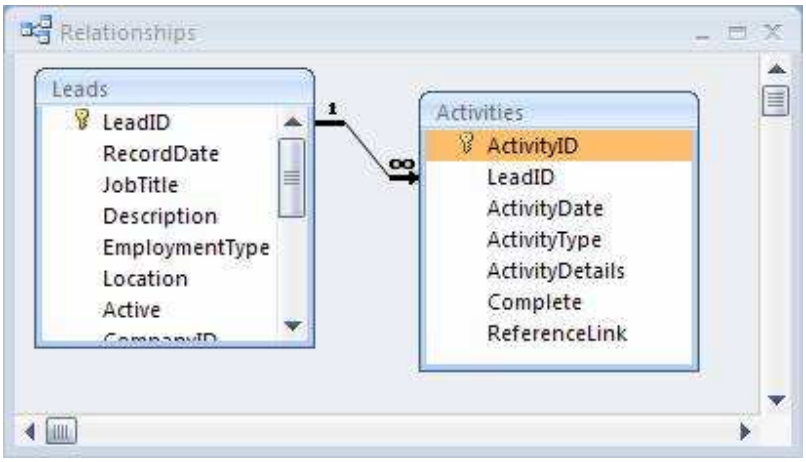

Figure 2.3 - Relationships between tables use corresponding fields such as record IDs to link data. In this example, the LeadID field links the Leads and Activities tables.

Using the LeadID value in the Activity table as a unique identifier, rather than something like the job title field, saves space and allows for additional report functions that you will see later. The nature of the Leads table also means that there is no one or even two fields that will be unique for each record. By using the number, the database knows exactly which Leads record each Activity corresponds to.

> *Sometimes it might be tempting to use a naturally unique value such as a Social Security Number or credit card number as a primary key, depending on what type of database you're building. I cannot stress this enough - NEVER use confidential information as a table key. You cannot prevent users from viewing table keys if they wish, especially if they're referenced by every other table as in this database.*

This arrangement between the Leads and Activities tables is an example of how a relational database works. In a flat spreadsheet, you might see all of the Leads fields laid out and then repeating groups like "Activity1", "Activity2", "Activity3", etc.. There are problems with that approach. First, it limits the number of activities that can be entered to an arbitrary number. It's also very difficult to search on. If you wanted to see all of the activities that involved a resume being sent, for example, you would have to search each of the fields separately. The approach shown in

Figure 2.3, on the other hand, changes the direction of the data. The Activities table can hold an unlimited number of activities for each lead and uses the LeadID as an identifier for which Leads record is being referred to. It's also easily searched with a simple query.

In the bottom half of the table's design screen, you'll see some extra settings for each field. When adding fields to a table, you should pay particular attention to settings such as Field Size, Input Mask, Default Value, the validation settings and Required. Proper use of these settings will provide the controls needed to help ensure that data is accurately entered. I provide more information on these properties in the next chapter and you can get more information on any of them by placing the cursor in one of the setting boxes and pressing F1.

Types of Relationships

If you select Relationships from the Access menu, it will bring up the relationship diagram for the database like the one shown in Figure 2.4. In Access 2007 / 2010, select the Relationships option from the Database Tools tab on the Office Ribbon. In Access 2003 and prior versions, it's available from the Tools menu.

This is an interactive tool that you can use to view and manage relationships between tables. The relationship diagram for JobSearch 2010 might look complicated until you remember some of the things I've mentioned so far and notice relationship lines between ID fields in one table and corresponding fields in another. The link between Leads and Activities is one of these.

Many of the relationships travel in the other direction. Reference tables such as Companies or Contacts use an AutoNumber as a primary field and the Leads table stores that value to refer to a specific item.

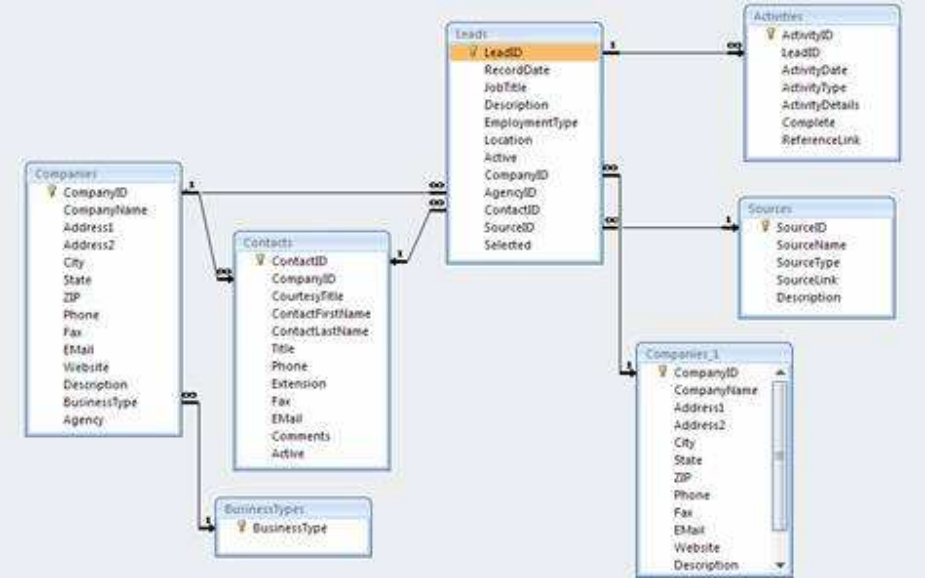

Figure 2.4 - The relationship diagram for the JobSearch 2010 sample database.

At first, this diagram might also seem to contradict what I said about making data entry easier. Another aspect of a relational database, however, is that it can contain data that is entered once and then referenced many times. The Companies and Sources tables in Figure 2.4 are good examples of this. The Companies table is a reference or *lookup table* for the information related to companies on various leads. By putting this information in a separate table from the job leads, the company information can be entered once and then referenced as needed. This is a lot better than having to enter a company name over and over again, hoping that it's always spelled right and then trying to retrieve all information on all leads related to a specific company.

By looking at the relationships between these tables, you can see how these relationships are defined. Double-clicking on a blank space within the diagram or on an existing relationship line will bring up the Edit Relationships screen where you can view the settings. In the relationship between Leads and Activities, the **1** and ∞ symbols above the line indicate a One-To-Many relationship between the tables. This means that for every job opening listed in Leads, there can be many activities recorded.

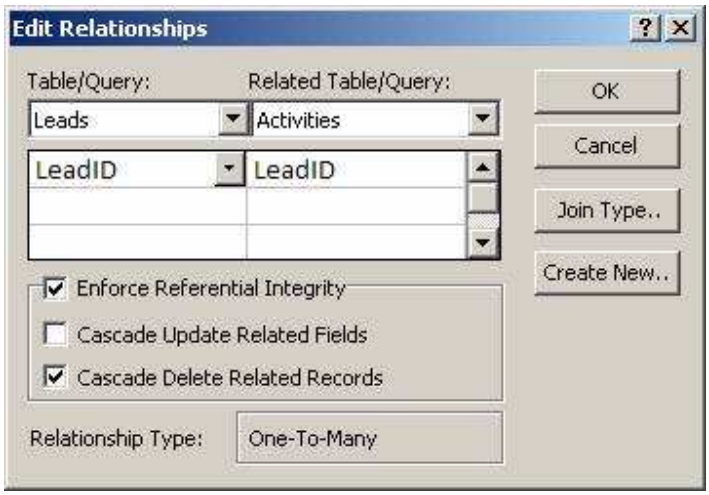

Figure 2.5 - The Edit Relationships dialog

Back in the relationships window, you'll also notice that the direction of the arrow points to Activities. If you click on the Join Type button on the Edit Relationships panel, you can see that the relationship is set to show all records from Leads and then any records in Activities that match those records. This will pull records from Leads even if there are no records for that job lead in Activities. It is referred to as an Outer Join. The first option on the Join Properties panel would pull only the records where the value for this field is present in both tables and this is called an Inner Join. In this case, job leads with no activities would not be listed and this might or might not be what you want. You'll see the importance of using the proper join type when you start designing queries to work with your table data.

A one-to-many relationship might be the most common type used but there are other types. A one-to-one relationship is formed when each record from the parent table can have no more than one corresponding record in the child table.

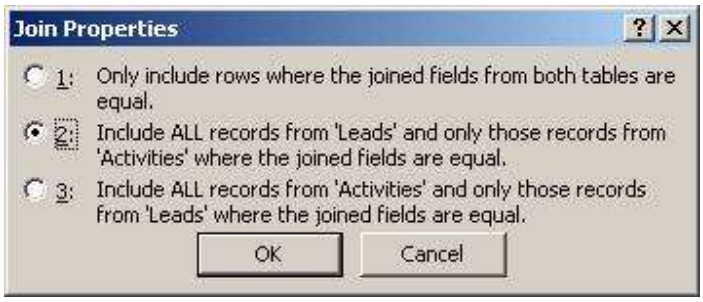

Figure 2.6 - The Join Properties dialog

The job search database does not have an example of this but one example would be a Human Resources database where the employee address and contact information is split off into a separate table from the main employees table as shown in Figure 2.7. This might be done to isolate the contact information for security purposes or simply to reduce the size of the main table.

In order for the database to recognize the one-to-one relationship in this example, the employee and contact information tables would need to link on an Employee ID field. This ID field would be the primary key and an AutoNumber in the employee table. In the contact information table, the corresponding field would be a regular numeric data type but would be set to prevent duplicate values so that a specific employee ID could only be used once in the table. The easiest way to prevent duplicates in the field, in this case, is to select the field as the Primary Key but you can also set a non-key field to be unique as you will see later. When you created the relationship between the two tables, Access would recognize that the employee ID field was unique in both tables and see it as a one-to-one relationship.

Sometimes, a third type of relationship, a many-to-many relationship, is needed. Again, it's not used in the job search database but an example would be a student database tracking class registrations. A hypothetical relationship diagram is shown in Figure 2.8. For each class, there will be many students and each student will take more than one class. This is a many-to-many relationship.

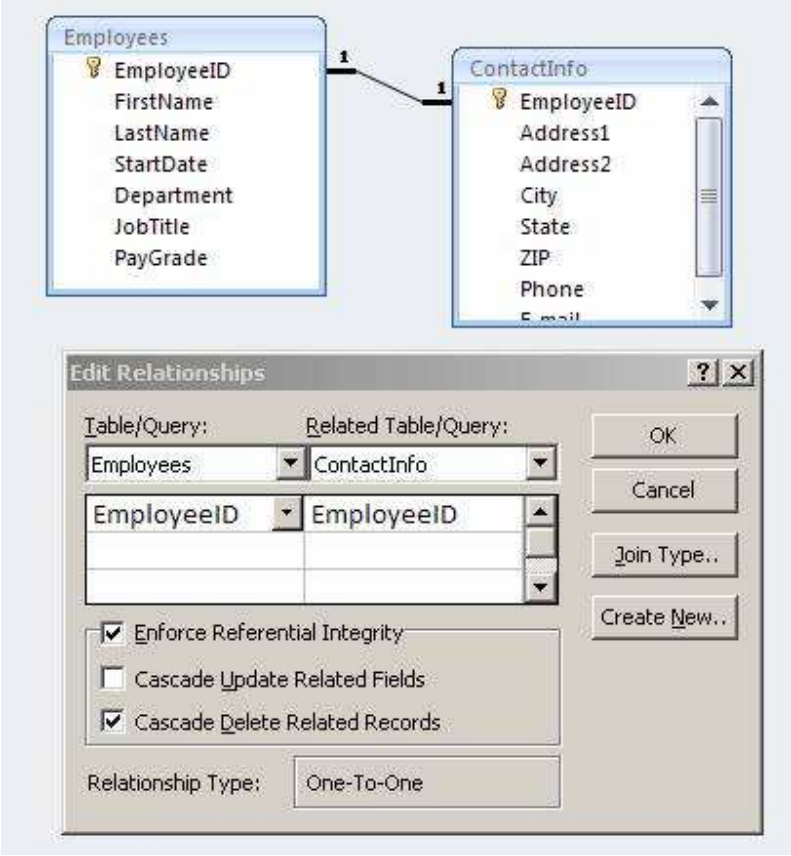

Figure 2.7 - Another example of a relationship between parent and child tables.

These relationships are not directly definable in Access so a third intervening table that breaks the relationship down into two one-to-many relationships is needed. This table is also known as a cross-reference table. In this example, a class registration table includes fields for the Student ID and the Class ID as well as any other information pertinent to that student's registration in the class. This table then links to both the student and class tables on those fields.

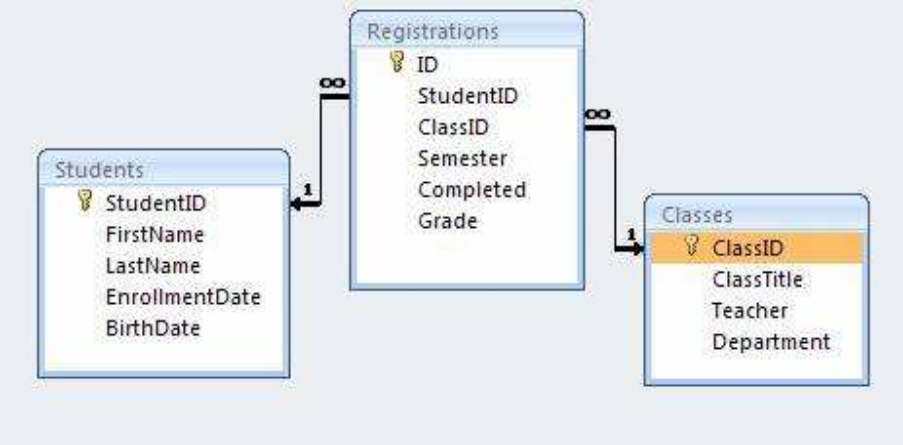

Figure 2.8 - A many-to-many relationship with an intervening table.

Referential Integrity

In the bottom half of the Edit Relationships box, you'll see a section dealing with something called *Referential Integrity*. This is a method for ensuring that if data from one table is supposed to match data in another table, then any changes are made on in both tables.

Figure 2.9 - Referential Integrity options

An example of this would be if one of the records in the Companies table was deleted. If there were job leads recorded for that company, this could cause a problem. Without referential integrity, you would be left with job leads but no company information. It would get even worse if a number of companies were deleted at once. With referential integrity activated, Access protects records and primary keys that are referenced by records in other tables. If you try to delete a record that is referenced by another table, you might get a message like the one in Figure 2.10. On the other side of the example, it would not be possible for someone to enter a non-existent company number in the Leads table because the database would not find a corresponding company in the Companies table.

Figure 2.10 - Referential integrity prevents the deletion of records that are referenced by other tables.

Of course, in some cases, you do need to be able to delete or change records, even if they are referenced in other tables. The two options under referential integrity for 'cascade' updating and deleting of records address this. Using cascade update, if a record's primary key is changed, that value is also changed throughout the database in any records that reference it. If cascade delete is active, then instead of getting the message shown in Figure 2.10, you'll get a message informing you that you are about to delete records in a related table and confirming that you wish to proceed. If you select Yes, then any records in other tables relating to that record would be deleted. In the case of the relationship between Leads and Activities, if a job lead record was deleted, a cascading delete would delete any activities associated with that lead.

The options for referential integrity are not automatically appropriate for every situation. Always take the nature and needs of the data being stored into account when you set options for tables and relationships.

Remaining Tables

In addition to the tables already mentioned, the job search database has others that are used for references and one or two interesting features.

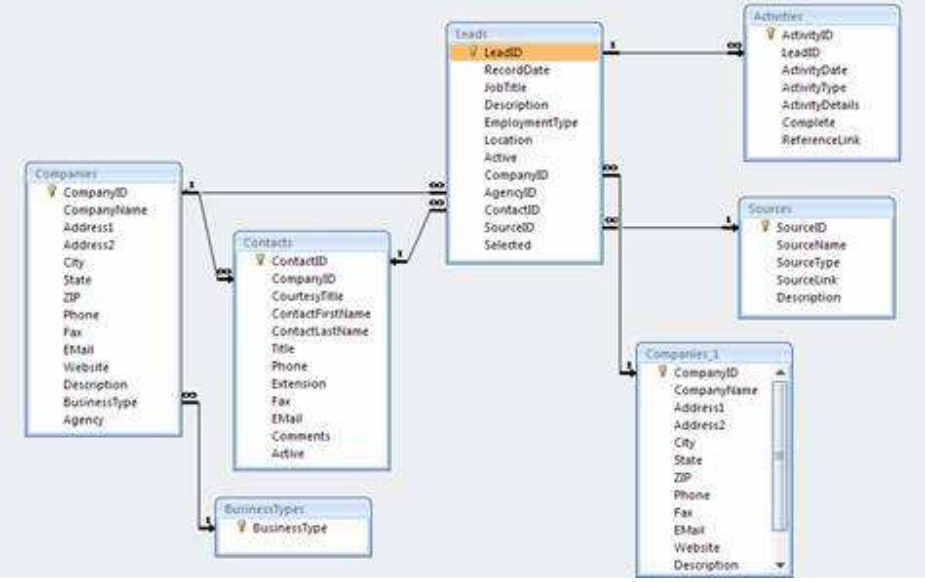

Figure 2.11 - JobSearch 2010 relationship diagram.

The Sources and Contacts tables act much like the Companies table, holding reference information as indicated. Sources stores a few details on lead sources such as online job boards and newspapers. It's linked directly to the Leads table with a one-to-many relationship so a Source reference can be stored with each lead. These tables are examples of how information that doesn't directly relate to the Leads table primary key is moved to a separate table for lookup as needed.

BusinessTypes is a basic lookup table that supplies business categories to the Companies table. Again, each category might be used an indefinite number of times so it has a one-to-many relationship to Companies. This table doesn't eliminate any information from Companies as the Business Type description is still stored. Instead, the table provides a list of types for the user to select from and *limits* the entry in the Companies table to those types, eliminating errors so that searches are more accurate.

The unique feature is the double listing of the Companies able under "Companies" and "Companies_1". These are actually the same table, just with a separate reference name in the relationships diagram. The reason for this is that the Leads table includes two fields that can hold a company reference; CompanyID and

AgencyID. This enables the user to record leads that are being pursued through a placement or recruiting agency. Since it would be exceedingly rare to have more than two companies that the job seeker would have to work with on any one job lead, I decided to include both fields in the Leads table rather than doing a separate table for the companies associated with a lead. By having both fields related to the Companies table, both can use the table as a lookup but it does require separate references in the Relationships window.

Also notice that the Contacts table is linked to both the Companies and the Leads table. The Leads table stores a primary contact for a job lead that might come from either the company or the agency and the Companies table links to it so that all contacts can be assigned to a specific company. The Companies-Contacts relationship would be the most obvious relationship but being able to link a table to multiple other tables provides flexibility in the database.

Database Normalization Overview

Now that you've seen an example of the rules in action, here are the first three normalization rules with some clarification. Normalization rules are referred to as *normal forms* and references to them are often abbreviated with the rule number followed by 'NF'.

First Normal Form (1NF): All fields contain just one value and there are no repeating groups within the table.

The first part of this rule is sometimes referred to as *atomicity*. Each field value within a table should be *atomic* meaning that, just like an atom, it can't be broken down into anything smaller and still be useful. An example of this would be a company address. When designing the table to hold company addresses, you *could* have a single field that held the city, state and zip code and your database might work okay until you decided that you wanted to search for companies by city or sort on the state in a report. You wouldn't be able to because these values would be part

of a larger value rather than having been stored separately. That's why you will usually see separate fields for the three values.

Depending on the database and what it's being used for, there might be some gray areas. An example would be a person's name. In most databases, you would want to split it into at least two fields; first and last, so that you could sort or search on either name. In some special cases, however, if you know that you're never going to need to do that, you could decide to keep the entire name together in one field.

There is a certain thing as over-normalization, where the normal forms are carried too far and no longer provide a benefit. An example would be if you decided to break up the street address into different fields, i.e. the street number, street name, suite number, etc. There is usually no clear benefit to this, it would result in a lot of blank fields (What if there is no suite number?) and it would force you to refer to more fields in order to construct an address. This is where a certain amount of judgment is required when normalizing data.

The second part of this normal form is that there are no repeating groups within a table. The Leads and Activities tables were used as an example earlier. In Excel or another denormalized environment, the activity items might be recorded in multiple numbered columns or groups of columns but this causes problems such as limiting the number of items that can be added to one order. The proper way to do this in Access is to change the direction of the data by having an Activities table with a record for each activity and a LeadID number that references the order that item is part of.

Second Normal Form (2NF): The requirements of the First Normal Form have been satisfied <u>and</u> all fields within the table are solely dependent on the entirety of the table's primary key.

Notice that the normal forms are cumulative. This requirement cannot be satisfied unless the values in your tables are atomic and repeating groups have been eliminated as mentioned in the First Normal Form.

The Second Normal Form simply means that every table must have a specific subject and every field within the table must relate directly to that subject. When evaluating a table against the rules of database normalization, that subject is what determines the field or combination of fields that would serve to uniquely identify the record. This is the primary key. In the example database, I often use AutoNumbers as primary keys for the purpose of linking tables but the normalization rules were not written with those in mind.

For example, in the demonstration database most job leads will have a company associated with them and if the table wasn't using an AutoNumber as the primary key, the company would be *one* of the fields that would be part of a unique identification for the job lead, probably in addition to the Job Title and the Date. These three fields would make up the basis of the job lead. While the company identifier is stored in the Leads table, there is no reason to store the company address or phone number in the same table because that information does not relate *directly* to the job lead itself but only to the company. It is supporting information but not primary information when referring to the job lead. If you were applying to more than one job at the same company, you would not want to have to store the same company address over and over again in different Lead records. This would be duplication and could lead to errors when the information wasn't copied correctly. Instead, a separate table is built for the company information and only the company ID number is stored in the Leads table. The Leads table then links to the Companies table to get the information as needed.

The primary goal of this normal form is to *avoid duplication of data.* If you find that you are storing the same information more than once in your database, you need to ask why. Information that is stored in multiple places takes up space and is harder to update consistently.

Third Normal Form (3NF): First and Second Normal Forms are met <u>and</u> all non-key fields within a table are mutually independent of each other.

This means no calculated fields or any other instances where the changing of one field will require a change in another.

In the example of a customer order database, this would prohibit the practice of having a total field that multiplies the number of items ordered by the item price. This type of field is not needed at the table level because it's easy enough to include as part of a query or data entry form so that it's available when it's actually needed. If you include it in the table, that means: a.) if the fields it's dependent on are ever changed, the field *must* be updated even if you don't need the information at the time which can affect the performance of your database or b.) the field is not updated automatically in which case it is not reliable. Calculated fields also take up additional space within the database for values that can easily be calculated as needed in a query or on a form or report.

Having said this, Access 2010 introduces calculated fields within tables. Microsoft's reasoning is that this provides a central location for expressions that might otherwise be duplicated on different forms and reports. Another reason is to increase the compatibility of Access with Microsoft SharePoint, a system used to share files and web-based information over networks. While I understand the logic here, I still believe that it is inherently poor design for a relational database because of the reasons stated above, especially when there are alternatives such as queries that only perform the calculation as needed. You'll learn about queries in an upcoming chapter. Nevertheless, Access will allow you to do a lot of things that fall outside best practices in database design and it is ultimately your responsibility to learn as much as you can and weigh all the options when designing your database.

When learning the rules of database normalization, you might hear it stated that every field in the table must provide information about "the key, the whole key and nothing but the key". This comes from E.F. Codd's definition of the Third Normal Form and it's a helpful guide to applying the rules in different situations. Some students even remember it as:

> "… the key, the whole key and nothing but the key, so help me Codd."

You probably have a basic understanding of these rules at this point but don't be surprised if it takes some practice on your part to learn how to apply them when designing your own databases.

Conclusion

Microsoft's support site offers a couple of very good articles which detail the rules of normalization and provide more examples. For more information, please refer to the Suggested Reading list at the end of the book.

As I listed the tables in this database, you might have found yourself thinking that it looks like a lot of data entry. If the tables were actually intended for data entry, that would be true. This is why I almost never allow users to enter data directly into the tables; that's what forms are for! In an upcoming chapter, you'll see how properly designed forms use the relationships between the tables to bring much of the data entry work together in one place. They can automatically supply many of the values required in the tables and include interesting features like drop-down lists and default values that eliminate a lot of typing and help to ensure correct data entry.

Next, I'll go into greater detail about how you can create tables like the ones you've seen here.

You have reached the end of this free preview of "Microsoft Access for Beginners". If you've found this book to be useful, you can find the rest of the book online at the following sites:

Amazon.com (Kindle)
http://www.amazon.com/Microsoft-Access-for-Beginners-ebook/dp/B0061RK6VO
Barnes and Noble (Nook)
http://www.barnesandnoble.com/w/microsoft-access-for-beginners-andrew-comeau/1114060042
Scribd.com (Adobe PDF)
http://www.scribd.com/doc/118841746/Microsoft-Access-for-Beginners

About the Author

Andrew Comeau is a Microsoft-certified programmer based in Ocala, Florida. He has been working with Microsoft Access since 1997 and provides independent consulting with that and other technologies. More information is available on his websites at Drewslair.com and AndrewComeau.com.