



#### Copyright © 2008 KSU Dept. of Information Technology Services

This document may be downloaded, printed or copied for educational use without further permission of the Information Technology Services Department (ITS), provided the content is not modified and this statement is not removed. Any use not stated above requires the written consent of the ITS Department. The distribution of a copy of this document via the Internet or other electronic medium without the written permission of the KSU - ITS Department is expressly prohibited.

#### Published by Kennesaw State University – ITS 2008

The publisher makes no warranties as to the accuracy of the material contained in this document and therefore is not responsible for any damages or liabilities incurred from its use.

Microsoft product screenshot(s) reprinted with permission from Microsoft Corporation.

Microsoft, Microsoft Office, and Microsoft Access are trademarks of the Microsoft Corporation.

# **Information Technology Services**

# Access 2007 Level 1

# **Table of Contents**

Objectives	
Overview/Introduction	7
What is Access?	7
Why would I use Access?	
Overview of the Changes for Access 2007	9
The Access 2007 Interface	9
The Office Button	9
The Ribbon	10
The Quick Access Toolbar	11
The Access 2007 File Format	
Macro Security Warning	
Overview of the Access Database Objects	14
The Objects	14
Table	14
Form	14
Query	14
Report	15
Macro	15
Views	15
How the Access Database Objects Work Together	16
Database Design	17
Database Design Glossary	17
Data Modeling	18
Overview	18
Identify Entities	
Identify the Attributes of your Entities	20
Consider the Relationships between your Entities	21
Assign Primary and Foreign Keys	22
Apply the Rules of Normalization	26
Concepts You Need to Know	
Data Validation	
Expressions	
Calculated Fields	
Getting Started	29
Creating a New/Blank Database	
Object Type View	
Tables and Relationships	
Creating a Table	33

Template Tables	33
Creating a Table in the Design View	34
Assigning a Primary Key	37
Assigning a Composite Primary Key	38
Creating a Relationship	
Introduction to Forms	42
Creating a Form with the Form Tool	42
Sample Small Business Database	43
Create a New Database	43
Create the Tables	43
Customer Table	43
Product Table	45
Order Table	45
Order Details Table	47
Inventory Table	47
Create the Relationships	48
A Final Word	48

# **Objectives**

Upon completing this classroom training workshop, the student should be able understand the following:

- What is Access?
- Why should I use Access?
- Changes for Access 2007
- Access Database Objects
- Relational Database Design

Upon completing this classroom training workshop, the student should be able to perform the following tasks:

- Use Entity-Relationship Diagrams (ERD)
- Create tables in Access 2007
- Create relationships between tables in Access 2007

# **Overview/Introduction**

#### What is Access?

Access is part of the Microsoft Office suite of applications. It is a relational database management system (RDBMS). This means that not only does Access have tables, but it also has other database objects (forms, queries, and reports) that help you with managing the data in your tables. It also means that Access stores data in a relational manner. Access's main purpose is to record large amounts of information in a relational manner.

Access is a desktop database system. It is meant to be used in a small office environment. Access is not a database server and cannot respond to hundreds of queries per second like a server database.

Sometimes it is a good idea to compare Access to Excel to get an understanding of the difference between the two products. Excel is a spreadsheet application that is meant to be used to analyze small amounts of data, whereas Access is a database application that is meant to be used to store large amounts of data in a relational manner.

Many people start saving their data in a spreadsheet. A spreadsheet, or worksheet, has a user friendly row and column layout that most people understand. Data stored in an Excel worksheet in a row/column fashion is called a list. A list in Excel is very similar to a table in Access; the two are interchangeable. In fact, a list in Excel can be imported into Access as a table, and a table in Access can be exported to a list in Excel. A list in Excel can have up to 65,000 rows, and a table in Access can have millions of records (rows).

A worksheet in Excel has rows and columns, where a table in Access has records (rows) and fields (columns). An Excel workbook can only have 256 worksheets; an Access database can have 32,000 tables. The main difference between Access and Excel is that Excel does not have the capability to relate data in one list with data in another list. Access will let you relate data in one table to data in another table.

# Why would I use Access?

The main reason to use Access is to store information. You may have a small business and want to record the details of a sale, or just want to save information about your music collection.

You can use either Excel or Access to store your data. They both can perform many of the same database type functions:

- Create gueries to sort and filter data
- Perform calculations
- Create pivot tables and pivot charts
- Generate reports
- Data-entry forms to add, edit, view, or delete data
- · Create mail merges with Word
- Connect to external data sources
- Import data from external sources

The key to making a decision to use Excel or Access to save your data depends on the complexity or amount of data.

Is there more than one entity involved? An entity is a major subject of your data. Entities usually become tables in your database. If your data consists of names, addresses, and phone numbers of people, what most people would call an address book, each record (row) is about a person, or in other words, one entity. This type of data is called flat. Flat data can be easily recorded and managed in a list in Excel, or in one table in Access. So, if your data is flat and will be in the thousands of records, or less, Excel would be a good choice.

The need for an Access database really comes in when you have more than one entity in your data model and/or you will have hundreds of thousands of records or less. For example, using the address book example above, if you need to record the actions of people in the address book. Let us imagine that you will need to record a sale of merchandise and relate the sale to a person in your address book (we can call them customers if we want, they are still people), then you would have data consisting of two entities: a customer, and a sale. In this scenario, you would need two tables: a customer table, and a sale table. In the customer table, each record would have information about the customer, and in the sale table, each record would have attributes of the sale, such as, a date, time, tax rate, and the customer number to which the sale is related. The sale table would need to be related to the customer table. Excel does not do relationships.

We will cover more about relationships in the "Designing a Database" section of this booklet.

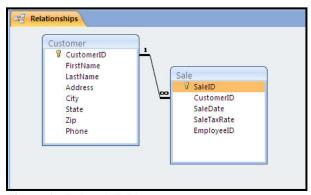


Figure 1 - Related Tables

# **Overview of the Changes for Access 2007**

#### **The Access 2007 Interface**

### The Office Button



The "Office Button", located in the top left corner of the window, replaces the "File" menu from previous versions of Office, and contains items that are common to all Office applications, such as "Open", "Save" and "Print".

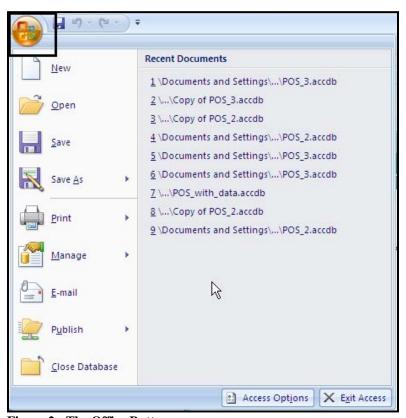


Figure 2 - The Office Button

When you roll your mouse over an Office Button item with an arrow displays a list of sub-items.

#### The Ribbon

The Menu Bar and Toolbars have been replaced by the Ribbon, a panel that contains functional groupings of buttons and drop-down lists organized by tabs. Each application in the Office Suite has a different set of tabs (with some tabs in common) that pertain to the functionality of that particular application. Each tab is further divided into groups (of buttons), such as the Font and Paragraph groups shown above.

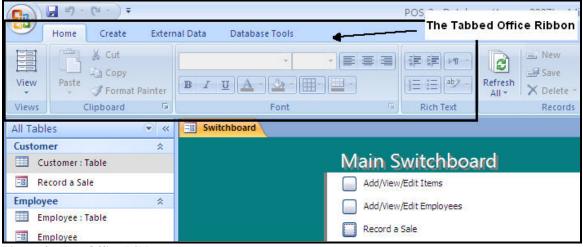


Figure 3 - The Office Ribbon

There are also "contextual tabs" that appear, depending on what you're working on. For example, if you have a table open in the "Normal" view, the "Table Tools" tab with a "Datasheet" tab is shown.

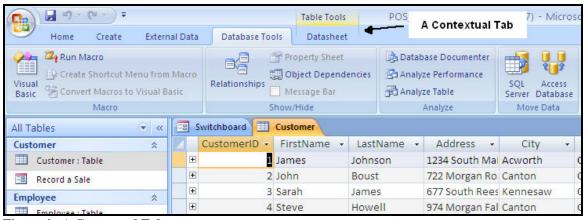
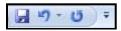


Figure 4 - A Contextual Tab

#### The Quick Access Toolbar



The Quick Access Toolbar is a small toolbar at the top left of the window that you can customize to contain the buttons for the functions that you use most often. In essence, it is the same as the toolbars from previous versions of Office with only the buttons you want to see on it.

To customize the Quick Access Toolbar, click on the drop-down arrow on the far right. You can add or remove items from the Toolbar, as well as show the Toolbar below the Ribbon, rather than above, and minimize the Ribbon.



Figure 5 - The Quick Access Toolbar

#### The Access 2007 File Format

Access 2007 uses an XML-based file format (ACCDB). Database files saved with Access 2007 will have the filename extension ".accdb". If you have databases in earlier formats, you can convert them to the new format.

To convert earlier Access databases to the Access 2007 file format:

- 1. Opening the database with Access 2007
- 2. Click on the "Office Button", and then click on "Save As".
- 3. Choose "Access 2007 Database".
- 4. Click "Save".

Some new features have been added to the Access 2007 file format:

- Multi-valued fields
- Storage of file attachments
- Encryption improvements
- SharePoint and Outlook integration

#### **Macro Security Warning**

When you open an Access database, you will see a "Security Warning" banner that states "Certain content in this database has been disabled", and there is a button labeled "Options" next to it.

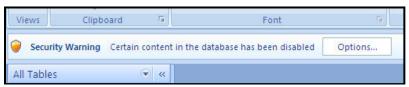


Figure 6 - Security Warning

This means that Access databases can contain code that can be executed on your computer. This code could be something that you created and installed in the database, or if you downloaded the database from an un-familiar source, someone else could have created it. The question is "Do you trust the content of the database?". If you do, click on the "Options" button, choose "Enable this content", and click "Ok". If you do not trust the database, choose "Help protect me from unknown content". Be aware that if you choose this, some of the features of the database may not work. Since we will be working with only databases that we create, we can choose to "Enable this content".

### **Overview of the Access Database Objects**

#### **The Objects**

Don't let the word "object" scare you, all it means is "thing". Access has database objects, or database things, which help you to store and manage your data.

**Table** – The table object is a storage type of an object. Your data is actually stored in a table. Tables are the database. If you have no tables, there are no reasons to have forms, queries, or reports. You can have up to 32,000 tables in your Access database. A table in the "Datasheet" view is very similar to a worksheet in Excel. The columns are called "fields" and the rows are called "records".

	Callbi	18	11			Save	ABC
Painter	В .	<b>I</b> <u>U</u>   <u>A</u> →			E E E Re	fresh	Spelling  More *
- G		For	nt	5	Rich Text	Records	
		Customer					
*		CustomerID -	FirstName +	LastName ·	Address	City -	State
	+	1	James	Johnson	1234 South M	ai Acworth	GA
*	+	2	John	Boust	722 Morgan R	Canton	GA
	+	3	Sarah	James	677 South Ree	es Kennesaw	GA
\$	Ŧ	4	Steve	Howell	974 Morgan Fa	al Canton	GA
	+	5	Tracy	Jackson	222 River Roa	d Canton	GA
<u>^</u>	±	6	Terry	Brown	777 Woodstoo	k Kennesaw	GA
~	+	7	Kyle	Evers	987 Canton Hi	g Marietta	GA
	±	8	Robert	Kemp	2348 Hickory I	R Kennesaw	GA
^	+	9	Laurie	Dhue	7262 Howell F	Kennesaw	GA
	1	10	Sidney	Lanier	8765 Morgan I	Ri Canton	GA
	+	11	John	Hampton	625 Lily Pad C	ir Mariette	GA
	Œ	12	Mary	Parks	1276 Mason V	/a Canton	GA

Figure 7 - An Access Table

Form – Forms are data-entry objects. Forms are usually the main interface between the user and the database. When entering data into a database, you should not enter data directly into a table; you should use a form to enter the data. A form will enable you to enter data into more than one table at once. Forms can also display calculated fields. Look in the "Concepts That You Need to Know" section for more information on calculated fields.

-	Employee	
	Employee	
	EmployeeID	
	FirstName	Steve
	LastName	Howard

Figure 8 - An Access Form

*Query* – A query is an output type of database object. A query is a custom view of your data. Queries enable you to ask questions of the data in your tables. A query will enable you to

combine data from many tables, up to 32 tables at once. A query is a type of filter that you can save with a name. You can have calculated fields in a query also. Calculated fields are very powerful tools in Access.

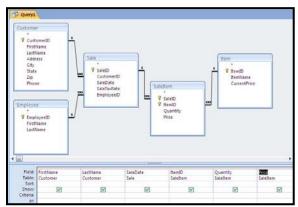


Figure 9 - An Access Query

**Report** – A report is another output type of database object. Reports are usually read-only, formatted for easy reading, and are meant to be printed. A report is usually based on a query. Whenever you run the report, the query is executed, and the data returned from the query is passed to the report for presentation.

*Macro* – A macro is a series of actions that are saved with a name, and can be run at any time the user wishes.

#### **Views**

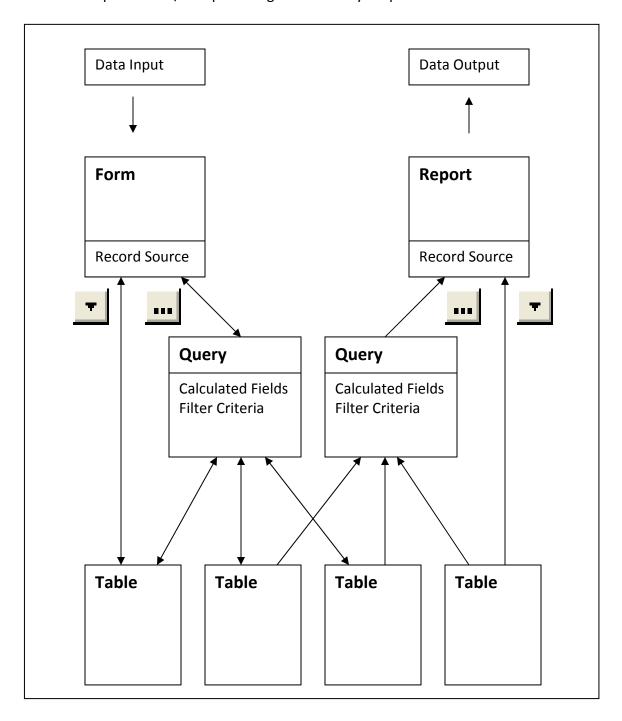
The database objects have different views in Access. They all have a normal view, which is the view that the everyday user of the database uses, and they all have a design view, which is the view that the designer of the database uses to modify the object. The user and the designer may be the same person, and they may not.



Figure 10 - Views of a Query

# **How the Access Database Objects Work Together**

When building an Access database, building the tables and setting the relationships come first. Then you build the forms so that you can enter data into the database. When you start to get data entered into the database you will want to see what is happening with your data so you build some queries and/or reports to get a summary output or a custom look at the data.



# **Database Design**

When building a database, it is important to have a plan. Create a plan, and then execute the plan. Before you open Access and start creating tables, it helps to have an understanding of the information that you want to record.

### **Database Design Glossary**

**Data Model** – A logical conversion of real-world things called entities into a relational database.

**Entity** – Information of importance about things involved in a business practice or process such as a customer, sale, item, or employee. An entity is a major subject of a data model. Entities usually become tables in your database.

**Associative Entity** – An associative entity is an entity that associates two entities that are related in a many-to-many type of a relationship. Associative entities usually become junction tables in your database. Associative entities can be abstract objects, like registration, or schedule.

**Attribute** – An attribute is a factor or property of an entity, like the last name of a person. Person is the entity; last name is an attribute of a person. Attributes will become fields in your tables.

**Primary Key** – A field or column in a table whose value uniquely identifies one record or row in a table. Each table in your database should have a primary key.

**Foreign Key** – A foreign key is a non-primary key in a table that relates a record (row) in one table to record in another table. Foreign keys are the basis of relationships.

**Relationship** – A relationship is the association between two entities. There re three types of relationships: many-to-many, one-to-many, and one-to-one.

**Entity-Relationship Diagram (ERD)** – A diagram that shows the entities, attributes, primary keys, foreign keys, and relationships, involved in a data model. ERDs are used in developing and refining database structure and design.

### **Data Modeling**

#### Overview

Data modeling is the practice of planning the logical foundation of your database. When designing a database in Access, data modeling is planning and building tables, fields, keys, and relationships. You do not need to be a computer scientist to build an Access database, although it does help to know the basics of relational database theory. It also helps to have a big whiteboard, or at least some large sheets of paper.

Keep in mind that during the process of data modeling you will need to always consider the needs of your organization. Why do you need a database? What information do you need to record? Think of the output that you will require from your database. That output has to come directly from data that is saved in your database, calculated from data that is saved in your database, or a combination of both. Consider the paper forms that you fill out now. These forms will give you an idea of the data that will be entered into the database. Consider the type of output that you will need. It may be a good idea to create a sketch of a report to aid in determining what information is desired on the report. A report in Access is a printable read-only type of output object. We will talk more about reports in detail later in this book.

If your database is going to be used by other people, or an organization, it is a good idea to include them in the data modeling process. Usually, three or four brains are better than one, right? Remember your group projects in college? Schedule them into a meeting just for the purpose of discussing the database. It also helps to use a whiteboard or large sheets of paper on an easel board.

#### There are five main steps to data modeling:

- 1. Identify entities
- 2. Identify the attributes of your entities
- 3. Consider the relationships between your entities
- 4. Assign primary and foreign keys
- 5. Apply the rules of normalization

In the following pages we will go through each of these 5 steps considering and sketching the details of a common model: a simple point-of-sale (POS) database system. If you own a small business, one of the most important sets of data that you will want to record is sales. If a

customer purchases an item from you they will expect a paper receipt which is a record of the sale which they can keep.

#### **Identify Entities**

Entities are the major subjects of your data model. Entities can be things, places, events, or concepts. A customer could be an entity, just as student, workshop, class, meeting, and sale, could all be entities. Entities will become tables in your database. A table is a database object where your information is recorded. Tables that describe entities are called base tables. The data modeling process starts with the entities and builds from there. I usually start by drawing or sketching the entities on a sheet of paper and then adding things as I go through the data modeling process.

If your information is about a person, like an address book, then the main entity is a person (sometimes called a contact or a customer), you would have a "Person" table. Every record in the "Person" table would be information about a person. As we consider entities, we will sketch them out on paper.

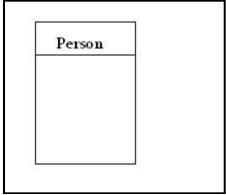


Figure 11 - One Entity

Yes, you can have a one entity database. A database that has only one entity is said to be "flat", there are no relations involved. You could also have many entities, it just depends on the complexity of the process that you are modeling. Usually each entity will become a table in your database.

Consider a point-of-sale (POS) model, let us imagine that you own a small business and you want to record information about sales. Whenever a **customer** purchases an **item**, a **sale** is made. Just from this simple statement we can easily deduce that there are three separate entities involved: customer, sale, and item. We may need to add some entities later, but these three will give us a good start.

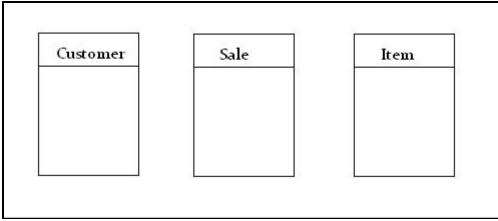


Figure 12 – Multiple Entities

#### **Identify the Attributes of your Entities**

Each entity in your model will have attributes that you will want to record. Attributes describe entities and will become fields in your tables.

In the point-of-sale model, a customer has a first name, a last name, an address, and so forth. A sale has a date on which it takes place, and a sales tax rate that is applied to the total. An item entity could have a name and a price.

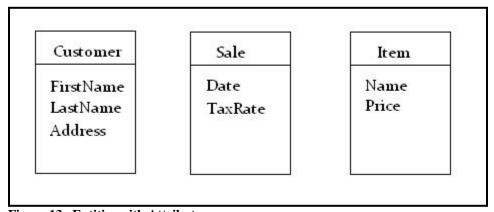


Figure 13 –Entities with Attributes

When thinking about the attributes of your entities, it is important to consider which of these attributes should be derived attributes. In Access, derived attributes are called calculated fields. Calculated fields are fields that are used in your database but not saved in a table; they are calculated from fields that are saved in a table and are calculated when they need to be displayed. These calculated fields are created by expressions (called formulas in Excel); more on calculated fields and expressions later.

An example of a derived attribute or calculated field is a person's age. You do not want to save the person's age in a table because their age will be different next year. You want to save their birth date, and when you need to display their age, you calculate it using their birth date.

An example of a calculated field in our POS database would be a line item total. The detail of a sale might be a list of items purchased. Each item could have a price. There may also be a quantity field, say if someone buys more than one of an item. Well if you have a quantity and a price, you will probably also want a line item total. You may also want a subtotal. The subtotal would be the sum of the line item totals. From the subtotal we can calculate a sales tax, and a total. Line item total, subtotal, sales tax, and total, are all calculated fields. We don't need to save this information in our tables, it just takes up space, we can calculate it when we need to display it.

In all of the previous examples, the entities will become tables, and the attributes will become fields in a table, and the derived fields will become calculated fields on a query, form, or report.

Another concern with attributes is "What are the allowable, or valid, values or for these attributes?" If you have ever heard the expression "Garbage in – garbage out", this is what they were talking about. For example, when recording a sale, if we want to record the employee that made the sale, if we allow the data-entry person to just type-in the name of the employee, it may be misspelled. Later, when we create a query to see the sales for that employee, the misspelled entries are not returned in the results. A solution for this situation is to force the data-entry person to choose from a list of valid employees when recording a sale. This list of valid entries is called a domain constraint. It contains the domain of valid values for this field.

This domain of all valid entries can be implemented in Access with what is called a lookup table. We will cover more about lookup tables in a section called "Data Validation".

### **Consider the Relationships between your Entities**

The entities in your database will be related to each other in one of three ways. It is up to you to think of exactly how they are related. Usually the way entities interact with each other in the real world is the way they should be related in your database. For example, a truck can carry many shipments. A shipment could be carried on many trucks (it may be a large shipment). So a truck entity could have a many-to-many relationship with a shipment entity.

There are three types of relationships: one-to-one, one-to-many, and many-to-many. The one-to-one relationship is not used very much compared to the others. You will find many many-to-many relationships. Access cannot represent a many-to-many relationship, so you will need to solve the many-to-many relationship into two one-to-many relationships using a junction table. Most of your relationships will be one-to-many, or many-to-many relations converted to two one-to-many relations. More on this as we go.

The address book model may only have one entity, a person (contact). This is perfectly acceptable; there would just be one table in the database. If there is only one entity, there will be no relationships. There is not another entity in which to relate a contact in a simple address book.

In the point-of-sale model, an item is related to a sale. A sale can have many items, and an item can be sold on many sales, so the sale and item entities have a many-to-many relationship. The many-to-many relationship between sale and item will have to be solved into two one-to-many relationships. You can do this by using an associative entity called SaleItem. The SaleItem entity will give you a way to associate a sale to many items, or an item to many sales. When a table is created to represent an associative entity, it is called an associative table; in Access it is called a junction table.

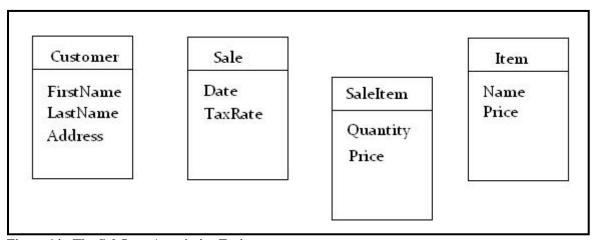


Figure 14 - The SaleItem Associative Entity

#### **Assign Primary and Foreign Keys**

Relationships are implemented with keys. A key is an attribute or field in a table that is a unique identifier. A key is unique to the record of which it is a part. A social security number is an example of a key. It is a unique identifier for a person. No two people should have the same social security number. A vehicle registration tag number is a unique identifier for a vehicle.

There are two basic types of keys: a primary key and a foreign key. A primary key is a field that is the main unique identifier for an entity/table. A foreign key is a field in table that is not the primary key, but relates a record in one table to a record in another table; it defines the association between entities, or tables.

Each one-to-many relationship has a primary key and a foreign key. The primary key is on the one-side of the one-to-many relationship, and the foreign key is on the many-side of the one-to-many relationship.

In the address book model, a social security number would be the best primary key to use. However, you may not have access to all of your contact's social security numbers. You could use an artificial, or derived, key like ContactID; more about artificial keys later.

When thinking of a primary key to use in a table, sometimes there may be an attribute that stands out as an evident choice for a key, such as a student id number. The reason the school created a student id number is to be able to uniquely identify a student in the first place. So if you can get the student to give his or her student id number, or if you have a way to get the student's id number, then it is a logical choice for a primary key of a student table. This would be called a real key, or a natural key; it has meaning in the real world.

In other circumstances, there may not be an evident candidate for a primary key. For example, in the item table, you may not have a real key for each item; so you might have to create your own artificial key. You could call it ItemID and use Access's auto-number data type so that each new record in the workshop table is a number that increments by one. In the following sketches, the primary keys are in bold font.

In the point-of sale model, there is a many-to-many relationship between a sale and an item, that is solved into two one-to-many relationships by using an associative entity called SaleItem. The Sale table, and the Item table, both need a primary key. The SaleItem table will need a primary key and two foreign keys.

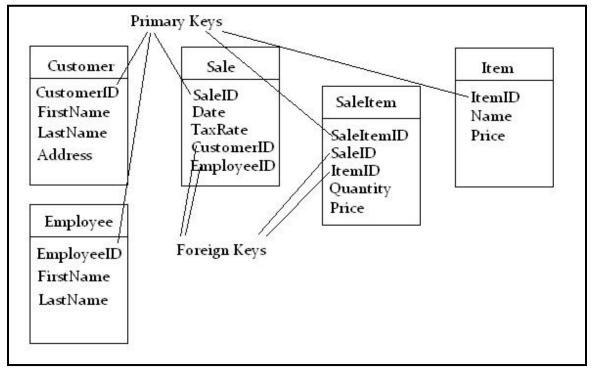


Figure 15 – POS Primary Keys and Foreign Keys

The following sketch shows how the foreign keys in the SaleItem table establish and maintain the relationships of these three tables. Each sale can now be related to many items, and each item can be related to many sales. Each sale can also be related to a customer and an employee.

At this point, there is something else to consider about this model. In the real world, we would not want a sales person to include an item on a sale more than once. That is what the quantity field is for; to show how many of one item a sale may include. There is a way to keep this from happening; we can use a composite key in the SaleItem entity. Consider the following:

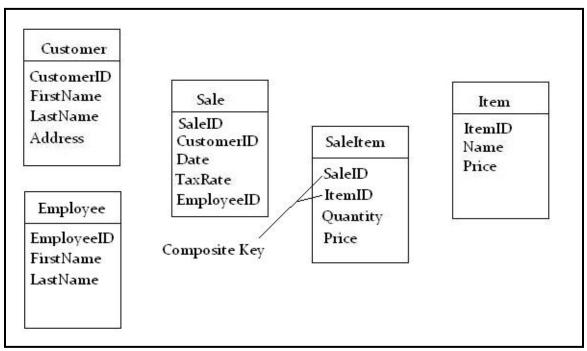


Figure 16 - Composite Key

A composite key will let you use two or more keys together and the combination of the keys will be the primary key. In this situation, if we tried to put the same item on a sale more than once, Access would complain that we were trying to use the same primary key twice. This could be considered a type of data validation; we are insuring that the same item does not appear on a sale more than once. Later in another chapter, when we are actually creating these tables, we will discuss how to create this composite key. It is really simple, you just select both keys and make both of them the primary key. In summary, these two foreign keys also act together as a primary key.

As you practice data modeling, you will think of new things that you will want to include in your model. Sometimes this can get out of hand (you can end up with 40 tables), sometimes it works very well. Let us think for a moment about these two models. What if I want to produce a report in the point-of-sale model that shows the sales for each employee that I have working the sales counter? I would need an employee attribute. Which entity would the employee

attribute fit the best? I would not want to put it in the Sale Detail entity because that entity is essentially a list of items purchased. The Sale entity would be the best place for the employee attribute. I want to record who made the sale, so the Employee attribute should belong to the Sale entity.

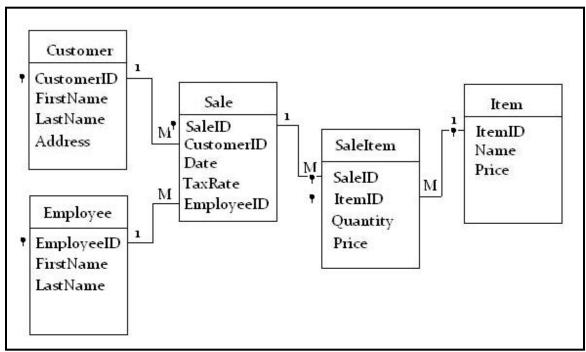


Figure 17 - Relationships

We need to apply a data validation technique to the EmployeeID attribute in the Sale entity. If we do not, the data entry person could easily make a mistake when typing in the name of the employee that made the sale. When I create a query and a report that pulls all of the sales by one employee, the sales with a misspelled employee will not be included in the results. Therefore, I do not get the correct results; garbage in, garbage out.

We can create another entity, which will eventually become another table in our database, called Employee. This Employee entity would have at the least a first name, and a last name attributes. This entity will become a table in our database that we can use to choose an employee that made the sale. The operative word here is "choose". When we build the Sale table, and the EmployeeID foreign key field, we can set the EmployeeID field as a lookup to the Employee table. This means that when the sale is recorded the person that is recording the sale can just choose the employee that made the sale from a list of all the employees.

#### **Apply the Rules of Normalization**

Normalization is the process of ensuring that your database adheres to relational database theory. The rules of normalization were developed to help you to validate the design of your relational database. After doing some initial design work, it is good to apply these rules to your plan.

We are only going to discuss the first three rules of normalization; for the beginner they are the most important. There are about seven rules, but for the scope of this document, we only need to discuss three. Your data model should conform to these rules. It does not always have to conform to all of the rules, except for the first one. We will talk about that a little later.

Examine your data model sketch and check to see that the following rules apply.

#### **The First Normal Form**

The first rule is the most important. If you followed the first few steps of the data modeling process, the chances are that your model already conforms to the first normal form. The first normal form states that you should break up your data to eliminate repeating groups, or what is called multi-valued attributes. This is what we did when we were trying to identify all of the main entities involved in your model. These entities usually end up as tables in your database. This process is why relational databases were invented.

#### The Second Normal Form

The second normal form states that each non-key attribute (non-key field) in a table should be directly dependent on the primary key for that table. That sounds simple enough. If you took care in determining the attributes for each of your entities, you should not have anything to worry about here.

#### The Third Normal Form

The third normal form states that you should not have any non-key attributes in a table that are dependent on a primary key in another table.

# **Concepts You Need to Know**

#### **Data Validation**

An important database design consideration is data validation. Invalid data can cause all of your queries to return incorrect or incomplete results. Have you ever heard the old saying, "Garbage in, Garbage out"?

Let's say that you want to be able to query your database and produce a report that shows the sales for each of your employees. When a sale is made, the salesperson that made the sale could enter their name so that sale can be identified as one that he or she made. If the salesperson misspelled his or her name, or entered it inconsistently, then the sales report queries will not pick-up that sale because the salespersons name is invalid. Why not have the salespersons choose their name from a list of valid entries for that field? If you create a table called "Employee", and fill it with a list of current employees, then you can set the employee field in the sale detail table as a lookup (sometimes called a drop-down list) to the employee table.

There are many data validation techniques, lookups are just one. Access has several methods of helping you to keep your data as valid as possible.

- Lookups
- Field Properties
- Composite Keys
- Input Masks

We will use these data validation methods later when we build the models that we designed earlier in this section.

### **Expressions**

Most people know of the power of formulas in Excel, Access has what Microsoft calls expressions. A formula in Excel is an arithmetic statement that can contain cell references. An expression in Access is an arithmetic statement that can contain fields from a table.

If you search the help section of Access for the word "expression", you will find many examples of how to use expressions. If you search for the word "formula", you will find nothing that will help.

Expressions can also contain functions. Like Excel, Access comes with many built-in functions. Any of these built-in functions, and any user-defined functions, can be used in expressions.

#### **Calculated Fields**

Not every piece of data that you consider when planning your database may belong in a table. Some information should not be saved, it should be calculated when it is needed. For example, a person's age; you should not record a person's age in your database, it will be incorrect next year. A person's age is dynamic, it changes every year.

When recording attributes of a person, you would want to record their birth date, not their age. The birth date does not change. Whenever you need to know or display the person's age, you calculate it with an expression.

There is no place to perform calculated fields in a table; you use calculated fields in forms, queries, and reports. This is one of the reasons of not entering your data directly into a table. You should use a form to enter data into your tables. We will cover forms in the Level 2 workshop.

# **Getting Started**

# **Creating a New/Blank Database**

To get started building a database in Access:

- 1. Click the "Start" button on the "Taskbar".
- 2. Point your mouse to Programs >> Microsoft Office >> Microsoft Access 2007.
- 3. The "Getting Started" task pane will open.



**Figure 18 - Getting Started With Access** 

4. Click on "Blank Database".

The "Blank Database" task pane will open.



Figure 19 - The Blank Database Task Pane

**Note:** Access will force you to name this database file and save it before you do anything with it. This is different from the other Office suite programs; they will let you work for hours and not save your work in a file. When you are entering data to an Access database, Access saves each keystroke as you type it. This is why you must save the file with a name and location from the beginning.

- 5. Enter the name for your new database in the "File Name:" box.
- 6. Click on the yellow file folder icon if you want to change the location to which your file will be saved.
- 7. Click the "Create" button.

Access opens your new blank database with a blank table in the Datasheet View. You can switch it to the "Design View" by clicking on the "Home" tab, and then the "View" button, and then click on "Design View".

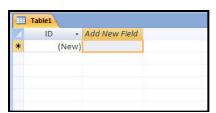


Figure 20 - New Table in Datasheet View

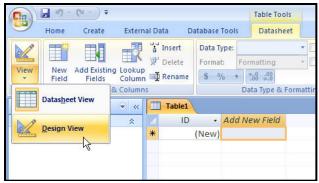


Figure 21 - Table in the Design View

You can close the table by clicking on the "X" button on the right side.

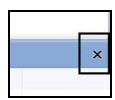


Figure 22 - The Close Table Button

# **Object Type View**

If you are more familiar with the "Object Browser" look of the earlier versions of Access you may want to switch the "Navigation Pane" to the "Object Type" view.

To switch to the "Object Type" view:

- 1. Click on the small black arrow at the top of the "Navigation Pane".
- 2. Choose "Object Type".

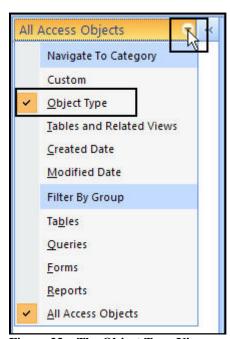


Figure 23 \_ The Object Type View

# **Tables and Relationships**

# **Creating a Table**

A table is a container for data. All of your data in Access 2007 will be stored in tables (columns are fields and rows are records). Tables are the most important database objects in Access. You can create as many tables as you need. Usually you will have a table for each entity in your data model; see the "Database Design" section of this document for more.

### **Template Tables**

There are some template tables that are provided with Access 2007. One of these tables may be close to your needs. If you choose to use a template table, you can modify it if you want.

To create a table from a template:

- 1. Click on the "Create" tab, and then the "Table Templates" button.
- 2. Choose one of the 5 template tables.

The new table will open in the "Datasheet View".

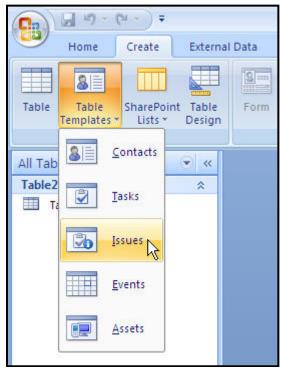


Figure 24 - Template Tables

To modify your template table:

- 1. Click on the "Home" tab, and then the "View" button.
- 2. Choose the "Design View".

### **Creating a Table in the Design View**

If there are no template tables that suit your needs, you may want to create a table from "scratch"; this is called creating a table in the "Design View".

To create a table in the "Design View":

1. Click on the "Create" tab, and then the "Table Design" button.

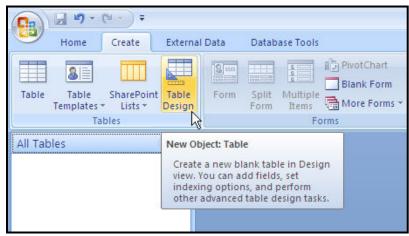


Figure 25 - Creating a Table in the Design View

A new table will open in the "Design View" and the "Design" contextual tab will open.

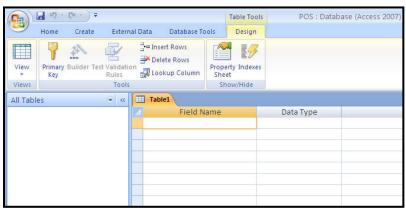


Figure 26 - New Table open in the Design View

The table "Design View" is divided into main 4 sections:

**Field Name** An identifier for that field (field names can be up to

64 characters).

**Data Type** The type of data that is allowed into that field

(column). The data type can be used as a data

validation technique.

**Description** Informs the user what to enter in this field on the

"Status Bar". Later, when you build a form for this

table, as the data-entry user is tabbing from

textbox control to textbox control, the description will be displayed in the status bar at the bottom left-hand corner of the screen. You can think of the

description as instructions to the data-entry user.

**Field Properties** Enables the designer to change the way the data is

stored or displayed. The field properties can also

be used as data validation tools.

- 2. Enter a "Field Name" for the first field in the table.
- 3. Press the "Tab" key.

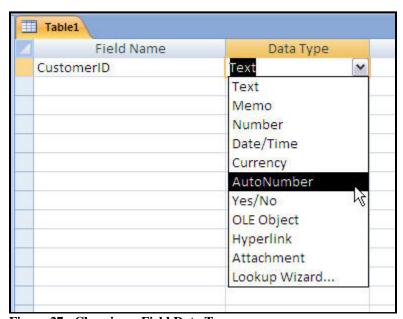


Figure 27 - Choosing a Field Data Type

4. Select a "Data Type" from the drop down menu.

Text	Text and/or numbers that will not be involved in a	
	calculation (Default)	
Memo	Lengthy text and numbers	
Number	Numbers that will be used in a calculation.	
Date/Time	Date or Time – Dates and times can be entered as	
	text, but you will not be able to use the built-in	
	functions that can manipulate	
Currency	Numbers that represent currency.	
AutoNumber	An integer that will automatically increment by one	
	every time a new record is created. This is good to	
	use as a derived key, if you do not have a real key	
	apparent.	
Yes/No	Check box—box is checked for "Yes', unchecked for	
	"No" – "True" or "False".	
OLE Object	Object, such as a picture	
Hyperlink	Hyperlink address	
Attachment	A computer file.	
Lookup	List box or Combo box that will enable the data-	
Wizard	entry user to choose a value for a field.	

- 5. Press the "Tab" key.
- 6. Enter a "Description". The "Description" is optional.
- 7. Press the "F6" key to move the cursor to the "Field Properties" area.

The "Field Properties" section changes to show the properties of the field that is selected. See below:

Table1 Selected	
Field Name	Data Type
CustomerID	Text
FirstName	Text
LastName	Text
Address	Text
City	Text
State	Text
Zip	Text

Figure 28 - Selected Field

Field Size	Sets the maximum number of characters that	
	can be entered in the field.	
Format	Controls the appearance of the data in the	
	field. Some Data Types, such as Date/Time,	
	allow you to select the format you want from	
	a drop-down list.	
Input Mask	Creates a pre-defined structure for data that is	
	entered into the field. For example, the input	
	mask for a phone number field would be:	
	(nnn) nnn-nnnn, where "n" is any single-digit	
	number.	
Caption	The name that is displayed as the column	
	heading in the datasheet view of the table.	
	This could be the same as the field name, or	
	different.	
Default Value	fault Value Data that is automatically displayed in the	
	field.	
Validation	An expression that limits the value entered	
Rule	into the field.	
Validation	An error message for invalid data.	
Text		
Required	Controls whether or not this field can be	
	skipped without entering data.	
Allow Zero	Along with the Required property, controls	
Length		
Indexed	Answering "Yes" creates an internal table that	
	contains the value of the field, and the	
	location of each record that contains that	
	value. Indexing speeds up sorting and	
	searching in a table.	

8. Enter any field properties necessary for the field.

# **Assigning a Primary Key**

A primary key is a field that uniquely identifies each record in a table. There will never be duplicate data in the primary key field.

1. With a table open in the design view, select the field which you want to set as a primary key.

2. Click the "Primary Key" button on the toolbar.

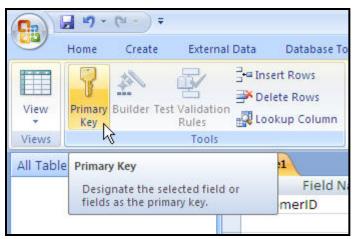


Figure 29 - Primary Key

## **Assigning a Composite Primary Key**

A composite primary key field is a combination of fields in which the fields together are unique.

To set a composite primary key:

- 1. With the table open in the design view, select one of the fields which will be part of the composite primary key.
- 2. Hold down the "Ctrl" key and select the other keys which will be part of the composite primary key.
- 3. Click on the "Primary Key" button.

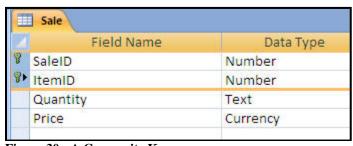


Figure 30 - A Composite Key

## **Creating a Relationship**

A relationship is a means of relating data from one table, to data in another table. Access 2007 supports three types of relationships: one-to-one, one-to-many, and many-to-many.

One-to-One	One record in a table has one related record in another table.	
One-to-Many	One record in a table has many related records in another table.	
	This is the most common type of relationship.	
Many-to-Many	Many records in a table have many related records in another	
	table. A third table is required to do this. The third table is called	
	a junction table. The other two tables should be related to this	
	junction table with two one-to-many relationships.	

**Important:** A relationship is implemented with a key pair: a primary key and a foreign key. Before tables can be related, one of the tables should have a primary key, and the other table should have a foreign key. In other words, the tables need to be designed so that they can be related. The data type of the foreign key must be compatible with the data type of the primary key.

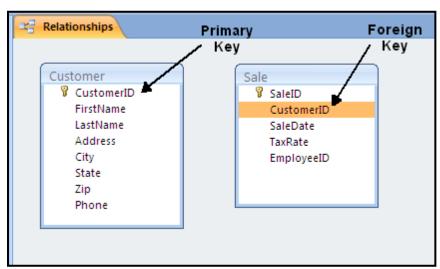


Figure 31 - Keys for Relating the Sale and Customer Tables

In the example above, the "CustomerID" foreign key in the "Sale" table will be used to relate a sale to a customer. When a sale is made, the "CustomerID" foreign key field will hold the customer number that will relate the sale to that customer.

To create a relationship (relate two tables):

1. Open the "Relationships" window by clicking on the "Database Tools" tab, and then the "Relationships" button.

The first time you use the "Relationships" window, the "Show Tables" window opens also; use it to add the desired tables. If you ever need to add more tables to the "Relationships" window, click on the "Show Table" button on the "Design" contextual tab.

- 2. Click on the foreign key, holding the mouse button down, and drag the foreign key over and drop it on the primary key to which you wish.
- 3. The "Edit Relationship" window will open. Check the option to "Enforce Referential Integrity".

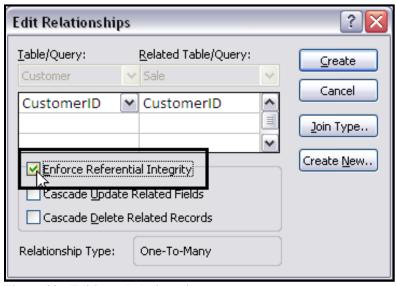


Figure 32 - Editing a Relationship

By enforcing the relationship, Access will not allow a value to be entered in the foreign key field that does not exist in the primary key field of the other table. It would not make sense to relate a sale to customer number 47 if there were no customer number 47.

4. Click the "Create" button.

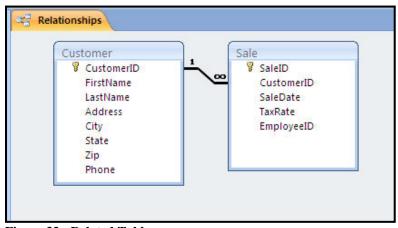


Figure 33 - Related Tables

The new relationship is indicated by a black line connecting the two tables. The "1" and the " $\infty$ " (the infinity symbol) indicate a one-to-many type of relationship. In this case, each customer can be related to many sales, but each sale can be related to only one customer.

5. Close the "Relationships" window.

**Note:** If you try to make changes to a field that is a key participating in a relationship, you may have to delete the relationship, make the change, and then re-create the relationship.

### To delete a relationship:

- 1. Right-click on the relationship (the black line).
- 2. Choose "Delete".

## To edit a relationship:

- 1. Right-click on the relationship (the black line).
- 2. Choose "Edit Relationship".

#### **Introduction to Forms**

Forms are used to add, edit, or view/display information in one or more tables at once. You should not enter data directly into a table. If you have a simple database that consists of one table, you might be able to get away with entering data directly into the table, but if you have more than one table, you should use a form.

### **Creating a Form with the Form Tool**

The "Form Tool" will create simple forms for us almost instantly. It is good to use when you want to create a form quickly. The "Form Tool" is smart; it can see the relationships to other tables and will create subforms when needed.

To create a form for entering data into a table using the "Form Tool":

1. Select the table that you want to build a form for by clicking on it one time. The example below uses the "Employee" table.

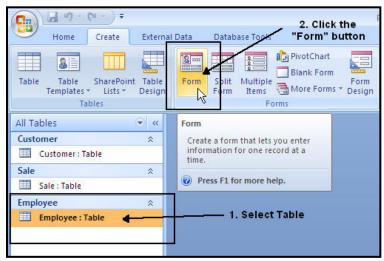


Figure 34 - Creating a Form with the Form Tool

2. Click on the "Create" tab, and then the "Form" button.

The "Form Tool" looks at the table that you have selected and creates a form for you automatically. If the table is on the one side of a one-to-many relationship, it will create a subform also.

Forms will be covered in greater detail in the Access 2007 Level 2 booklet and workshop.

# **Sample Small Business Database**

The following example is a database that records customer, order, and inventory information for a small garden supply company. This database could be expanded and made more complex, but, it is simplified for the scope of this course.

#### Create a New Database

Create a new blank database. See <u>Page 29</u>. We'll call this database Kennesaw Garden Supply.

#### **Create the Tables**

This database contains five tables: Customers, Products, Orders, Order Details and Inventory. The Order Details table contains line item detail about each order. The Inventory database tracks the supply of products on hand.

Each table is created in Design View. See <u>Page 34</u>.

#### **Customer Table**

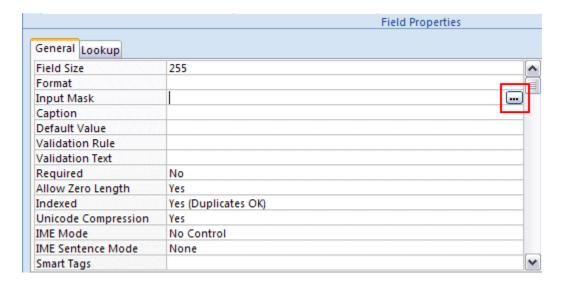
## Fields/Data Types

4	Field Name	Data Type
8▶	Customer ID	AutoNumber
	Customer Name	Text
	Address	Text
	City	Text
	State	Text
	Zip Code	Text
	Phone Number	Text

The Zip Code and Phone Number fields must have the "Text" Data Type, since these fields will contain both numbers and special characters.

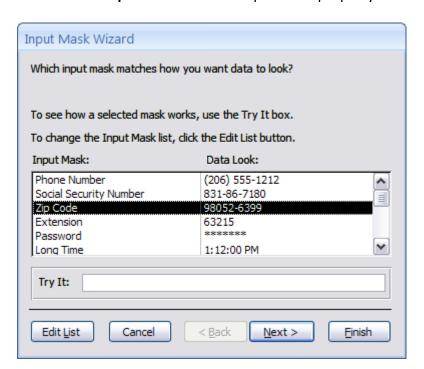
#### **Specific Field Properties**

The Zip Code and Phone Number fields both use an "Input Mask" to assist the user in entering data.



To create an Input Mask for a field:

1. Click the Ellipses Button in the Input Mask property.



- 2. Select the type of input mask that you want.
- 3. Click Finish. (Clicking "Next" gives other options that we don't need now.)

#### **Product Table**

#### Fields/Data Types

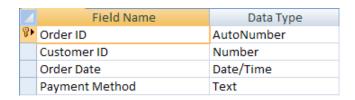
8	ProductID	AutoNumber
	Product Name	Text
	Price	Currency
	Reorder Quantity	Number

#### **Specific Field Properties**

None.

#### **Order Table**

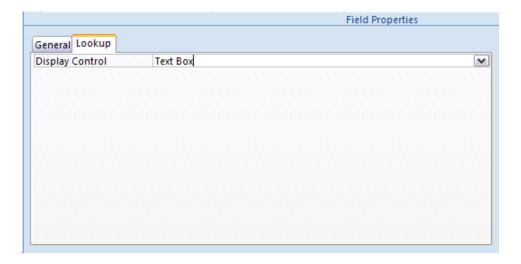
## Fields/Data Types



## **Specific Field Properties**

We will create a "lookup" for the Payment Method field that will allow us to select the payment method from a drop-down list. Since there will only be a few items in this list, we will manually add the items to the list, rather than creating a separate table of items.

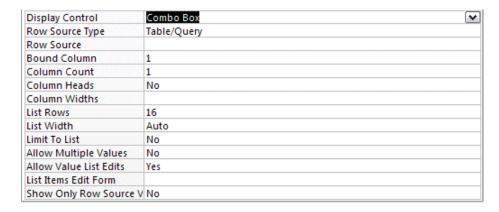
- 1. Click in the Payment Method field.
- 2. Click the **Lookup** tab in the Field Properties area.



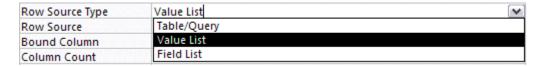
3. Click the down arrow in the **Display Control** property.



4. Select **Combo Box** from the drop-down.



5. Click in the **Row Source Type** property and select Value List from the drop-down.



6. Click in the **Row Source** property, and then click the ellipses button.



7. Type each payment method on a separate line, and then click OK.

### **Order Details Table**

The Order Details table is the junction table between the Order and Product tables, providing the necessary many-to-many relationship. The Order Details table also contains specific line-item details for each order.

## Fields/Data Types

4	Field Name	Data Type
8₽	Order Detail ID	AutoNumber
	Order ID	Number
	Product ID	Number
	Quantity	Number

# **Specific Field Properties**

None.

## **Inventory Transaction Table**

## Fields/Data Types

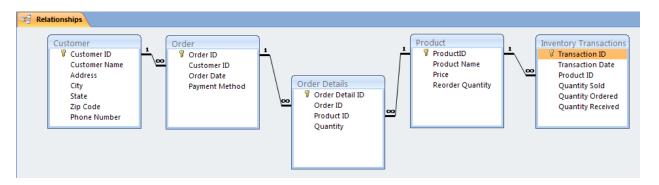
	Field Name	Data Type
8	Transaction ID	AutoNumber
	Transaction Date	Date/Time
	Product ID	Number
	Quantity Sold	Number
	Quantity Ordered	Number
	Quantity Received	Number

# **Specific Field Properties**

None.

## **Create the Relationships**

For detailed information on relationships, see <a>Page 39</a>.



The relationship between the Customer and Order tables is a one-to-many, based on the Customer ID field.

There is a one-to-many relationship between the Order table and the Order Details table, based on the Order ID field. There is a one-to-many relationship between the Product table and the Order Details table, based on the Product ID field. These two relationships combine to create the many-to-many relationship between the Product table and the Order Table. (One order can contain many products, and each product can be contained on many orders.)

There is a one-to-many relationship between the Product table and the Inventory Transactions table, based on the Product ID field.

#### **A Final Word**

You may want to experiment with creating Forms to enter data into the tables in this database, using the quick method described on <a href="Page 42">Page 42</a>. We will cover Forms in depth in the Access 2007 Level 2 course.