

The Easy Way Series

Access 97 Level Two



Access 97, Level Two
Software Learning Guide
Revision: AW972990817

ITRAIN MISSION STATEMENT

To facilitate the highest quality information technology training programs. To promise association members the finest professional services that facilitate improving their training.

CREDITS

Dave Murphy
Teresa Crone

TRADEMARKS

ITrain and We Help Trainers Train Better are trademarks of ITrain, the International Association of Information Technology Trainers. Training Express, You're On The Right Track Now!, Damar Group and Your Personal Computer Expert are trademarks of Damar Group, Ltd. All other trademarks are the property of their respective owners.

DISCLAIMER

ITrain makes no warranty, expressed or implied, with respect to the quality, correctness, reliability, currentness, accuracy, or freedom from error of this document or the products it describes. ITrain makes no representation or warranty with respect to the contents hereof and specifically disclaims any implied warranties of fitness for any particular purpose. ITrain disclaims all liability for any direct, indirect, incidental or consequential, special or exemplary damages resulting from the use of the information in this document or from the use of any product described in this document. Mention of any product does not constitute an endorsement by ITrain of that product. Data used in examples and sample data files are intended to be fictional. Any resemblance to real persons or companies is entirely coincidental.

COPYRIGHT NOTICE

This training guide and sample data files which may be used with it are copyright © with all rights reserved by ITrain. No part of this publication or data may be reproduced, transmitted, transcribed, stored in a retrieval system or translated into any language or computer language in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise without the written permission of ITrain.

ITrain - International Association of Information Technology Trainers
PMB 625
6030-M Marshalee Dr
Elkridge, MD 21075-5935

1.888.290.6200
603.925.1110 (fax)

itrain.org
member@itrain.org

Copyright © 1999 ITrain - International Association of Information Technology Trainers,
All Rights Reserved

About The Authors

Dave Murphy has taught thousands of students, novice-to-expert, to be more productive using personal computers. He has been certified by Novell, Inc. as a Certified NetWare Instructor and by WordPerfect Corporation as a WordPerfect Certified Resource®. His training expertise includes WordPerfect, Paradox, dBASE, Lotus 1-2-3, Quattro Pro, Windows, Novell NetWare, and over a hundred other popular software products. In addition to presenting college and business training programs, David is a sought-after public speaker for business development programs within both the academic and commercial forums, and he has standing columns in three business newspapers. David holds academic degrees from the Defense Language Institute, United States Air Force, The University of Maryland, and The Johns Hopkins University. After a career in the military, David is President of Damar Group, Ltd. in Columbia, Maryland, a corporation controlling training and publishing firms. He is the founder of ITrain, the International Association of Information Technology Trainers. He may be reached at 410.290.7000 or on the Internet at dave@dgl.com.

Teresa Crone has taught and worked with office related software since the 1980's. She has been ITrain and Damar Group, Ltd.'s lead trainer in the Microsoft Windows and Microsoft Office Suites since the products arrived on the market. Beside teaching all levels of Word, Excel, Access, PowerPoint, Mail, and Outlook, she teaches Quicken Deluxe, Internet and Hard Disk Management classes as well as DOS and Windows versions of WordPerfect, Lotus, Paradox, ACT!, and PageMaker. In addition, Ms. Crone has developed custom database and office applications for a wide variety of Damar Group, Ltd. clients. She is the most requested trainer at Damar Group, Ltd. Teresa is a charter member of ITrain, the International Association of Information Technology Trainers. She may be reached on the Internet at teresa@dgl.com.

Contents

Database Structure	1
Introduction	1
Field Structure	1
Record Structure	2
Field Data Types	2
Relational Database Management Systems	3
Field Structures And Relations	4
Relational Database Management	5
Introduction	5
Customers Table Structure	6
Products Table Structure	7
Orders Table Structure	7
All Orders Query Structure	8
All Orders Query Result	8
All Orders Report Structure	9
All Orders Report Result	9
Creating A Relational Database	10
Introduction	10
Creating The Customers Table	10
Modifying The Table Structure	11
Creating The Products Table	13
Modifying The Table Structure	14
Creating The Orders Table	14
Modifying The Table Structure	16
Viewing Relationships	16
Editing Relationships	17
One-To-Many Relationship	18
Customers Data Entry	18
Products Data Entry	19
Orders Data Entry	20
Form Navigation	21
Introduction	21
Creating a Command Button	21
Exercise	24
Introduction	25
Creating A Relational Query	25
Relational Queries	25
Adding Fields To The Query View	26
Adding A Mathematical Expression Field	26
Running The Query	27
Saving The Query	27
Relational Reports	28
Introduction	28
Creating A Relational Report	28
View The Finished Report	31
Saving The Report	31
Index	32

Database Structure

Introduction

Throughout this learning guide, it is assumed that the reader has a working knowledge of Access, and that he or she is able to create, query, and report single-table databases.

Computer database management systems combine pieces of data to create usable information. Most people are able to grasp the usefulness of a set of data quickly; however, for this data to be stored in computer format requires that it be separated into discrete chunks.

Take the following information for example:

David S. Murphy
Damar Group, Ltd.
9810 Patuxent Woods Drive
Columbia, Maryland 21046
(410) 290-7000
(Fax) 290-7790

You probably recognize this as a person's address, telephone number, and facsimile number. However, this "data" must be stored in a computer database as in separate chunks, called *fields*.

Field Structure

To enter Mr. Murphy's address information into a computer database management system, the required fields may be:

Name:	David S. Murphy
Company:	Damar Group, Ltd.
Address:	9810 Patuxent Woods Drive
City:	Columbia
State:	Maryland
ZIP:	21046
Telephone:	(410) 290-7000
Facsimile:	(410) 290-7790

This field structure is simple; however, it is not as flexible as you may need it to be.

For example, after many people have been entered into the database, it would be awkward to extract only those people who have the same last name, all the people named "Murphy."

It's difficult to extract the Murphys because each person's full name is entered into the same field, *Name*.

In this example, the same data may be more easily extracted:

First Name: David
Middle Initial: S.
Last Name: Murphy
Company: Damar Group, Ltd.
Address: 9810 Patuxent Woods Drive
City: Columbia
State: Maryland
ZIP: 21046
Telephone: (410) 290-7000
Facsimile: (410) 290-7000

With this more specific field structure, it's possible to extract all people where *Last Name* = *Murphy*, were the field *Last Name* equals *Murphy*.

Also, if the field structure separates the first and last names, it's possible to sort and alphabetize, the database list by last name or by last name and first name.

Record Structure

A record is a collection of fields. All of the completed fields for Mr. Murphy constitutes one *record*. Information about Ms. Jane Doe would constitute a separate record.

Within a record, each field must have a specific data type. All of the fields in the previous examples should be *text* data type fields. However, if the field structure included age or date of birth fields, then it would be appropriate to use *numeric* or *date* data type fields, respectively. Whenever mathematical calculations may be performed on data (such as quantities or prices), the field type should be set as a form of a number field.

Field Data Types

Access includes the following field data types:

Text
Memo
Number
Date/Time
Currency
AutoNumber
Yes/No
OLE Object
Hyperlink
Lookup Wizard

Non-numeric fields that contain only digits should be created as text fields. ZIP codes and phone numbers are examples of non-numeric fields that may include only digits.

Each of these field data types has specific uses. The data type controls the type of information that may be stored within the field. Text fields allow any characters (letters, digits, punctuation, etc.) to be entered. Memo fields allow paragraph-style formatting of any characters. Number fields allow only digits, numeric information, to be entered. Date/Time fields are preformatted for date or time entry. Currency fields automatically insert a currency symbol (\$). AutoNumber fields increment automatically; they are useful for sequential assignment for each record. Yes/No fields accept only a "yes" or "no" entry. OLE Object fields allow any object to be inserted into the field, such as a graphic picture or line drawing. OLE is the Microsoft standard for Object Linking and Embedding of linked objects.

Hyperlink fields allow links to another object, document, Web page or other destination. Lookup Wizard allows you to create a field which selects data from another table or from a set of values you can define.

The most commonly used field data type is *text*. Because the text-type field is so flexible, it meets the data entry needs for many circumstances.

If a field contains count, quantity, or dollar amounts use the number or currency field type as these fields may be used for mathematical operations and summations.

Relational Database Management Systems

Access is a relational database management system. The term *relational* implies that multiple tables of records may be *related* to one another in some fashion. Because the data tables may be related, operations may be performed that require information from multiple tables at the same time.

Relational databases allow data to be validated amongst the tables and preclude the operator from entering the same data more than once.

Take for example the following relational database structure:

<i>Table 1</i>	<i>Table 2</i>
CustomerID	CustomerID
Company	ProductCode
Address	PurchaseDate
City	Quantity
State	
ZIP	

Table 1 includes the customer's unique identification number and mailing information.

Table 2 includes the Customer ID, Product Code, Purchase Date, and Quantity information. As a customer returns to your store to purchase a product, the sales transaction is entered into Table 2. However, because both tables share the common field *Customer ID*, the tables may be related.

Once the tables are related, it's possible to ask (*query*) the database for all customers who have an address in the State of Maryland and have purchased a particular product.

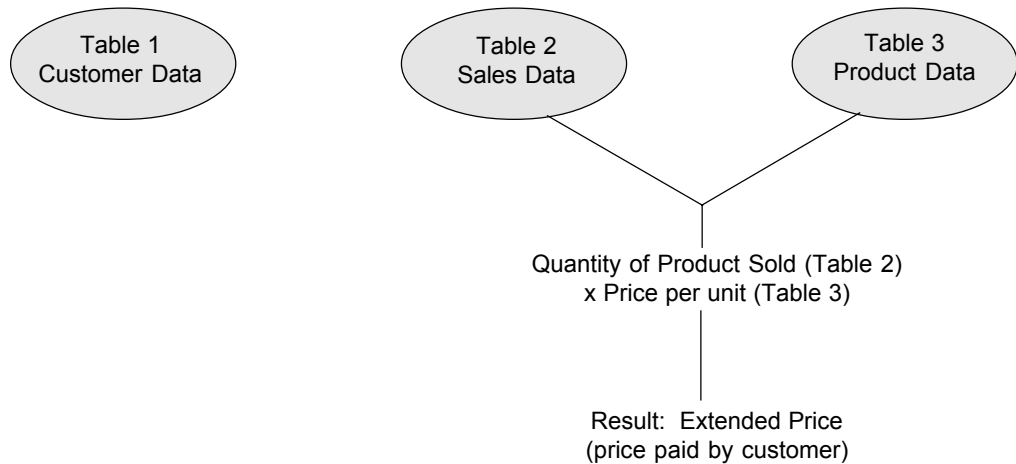
Relational database management systems also allow data to be validated amongst the related tables. It's possible to restrict Table 2's Customer ID field to allow only information that is already stored in Table 1's Customer ID field. This would prevent a sale being made to a customer who was not already in Table 1.

From a retail management perspective, this prevents erroneous sales data as well as preventing intentionally misleading data entry, especially if sales may be made only to approved customers, those who are listed in Table 1.

Field Structures And Relations

Correctly, only fields are stored within the database table. If multiple numeric or currency fields are used within a mathematical expression, the result would not be stored in the database, in a separate field. The result would display on the screen or print on the report only at the time of display or report execution. In other words, the mathematical expression is evaluated each time the result is requested at the screen or printer.

In the following example, three tables are related. Table 1 and Table 2 were discussed above, Table 3 includes product information such as Product ID and Price per unit.



Using all three related tables, it's possible to request mailing labels (a type of report) for all customers who have spent over \$500 within the last six months. The customer's address comes from Table 1, the total amount spent is the result of the multiplication expression, and the date period may be extracted from Table 2's Purchase Date field.

Relational Database Management

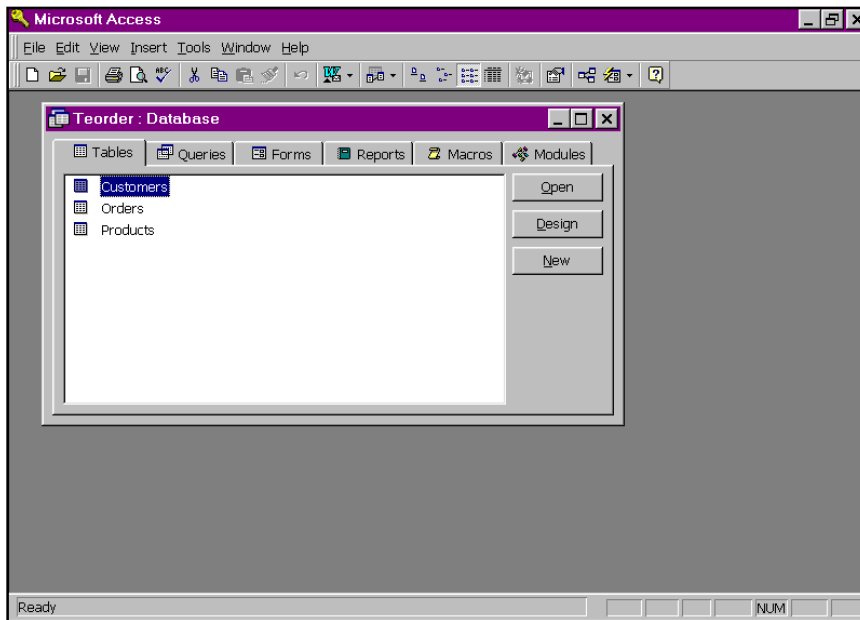
Introduction

In this exercise, you will use a relational database management system included in the database TEORDER to demonstrate the capabilities of related tables.

1. Click **Open Existing Database**
2. Click **More Files**
3. Click **OK**
4. Click **TEORDER**
5. Click **Open**

Your Instructor will direct you to the correct file path to open the database.

The TEORDER database includes three tables: Customers, Orders, and Products.

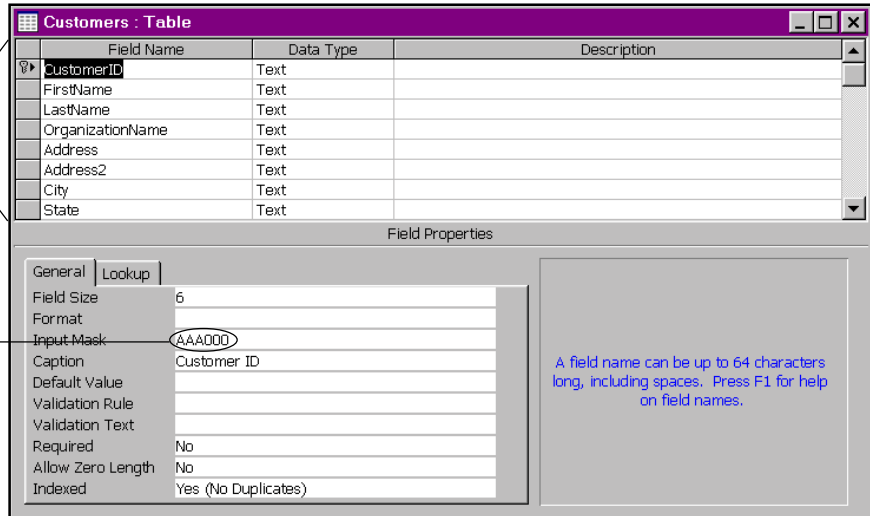


Customers Table Structure

The Customers table includes a key field, CustomerID, which ensures each customer record has a unique CustomerID entry.

1. Click **Customers**
2. Click **Design**

Field structure of Customers table



Input Mask

The input mask controls what characters (letters, numbers, or other symbols) may be entered into the field and the formatting restrictions.

The Customers table has the following structure:

<i>Field Name</i>	<i>Data Type</i>
CustomerID	Text
FirstName	Text
LastName	Text
OrganizationName	Text
Address	Text
Address2	Text
City	Text
State	Text
PostalCode	Text
PhoneNumber	Text
FaxNumber	Text
Note	Memo

Notice that since the PostalCode, PhoneNumber and FaxNumber fields may contain a dash or parentheses, they may not be designated Number fields where only digits are allowed.

Input Mask

The CustomerID field has an *input mask* of AAA000. This mask restricts the field to only accept input of data in the form of three letters followed by three digits (i.e. ABC123, BBS232, and CGA121).

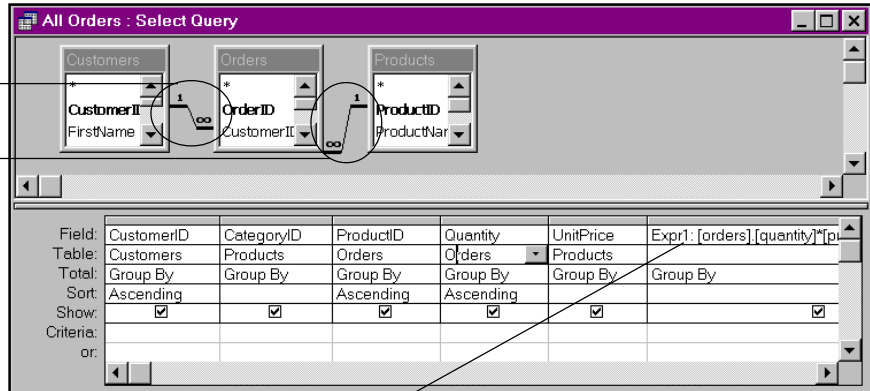
The input mask facilitates and controls data entry into the field to a predetermined format. In

All Orders Query Structure

1. Close the Orders table structure window.
2. Click **Queries tab**
3. View the design of the All Orders query.

Customers & Orders tables are linked in a one to many relationship

Products & Orders tables are linked in a one to many relationship



Mathematical expression:
 $[orders].[quantity]*[products].[unitprice]$

All Orders Query Result

When the All Orders query is run, the result includes the mathematical expression field: Extended Price.

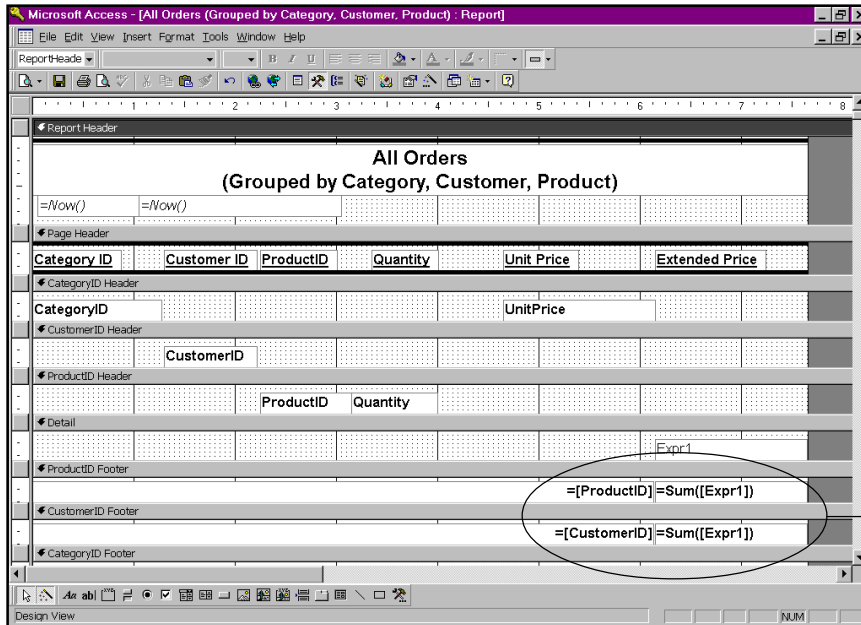
1. Click  **(Run)**

Extended Price: result of mathematical expression

Customer ID	Category ID	ProductID	Quantity	Unit Price	Extended Price
ABC123	Screw	EC	3	\$1.95	\$5.85
CGA121	Container	WT	10	\$95.42	\$954.20
WIG992	Screw	SCT	3	\$0.09	\$0.27
WTM333	Widget	SWB	1	\$0.05	\$0.05
WTM333	Container	WT	2	\$95.42	\$190.84

All Orders Report Structure

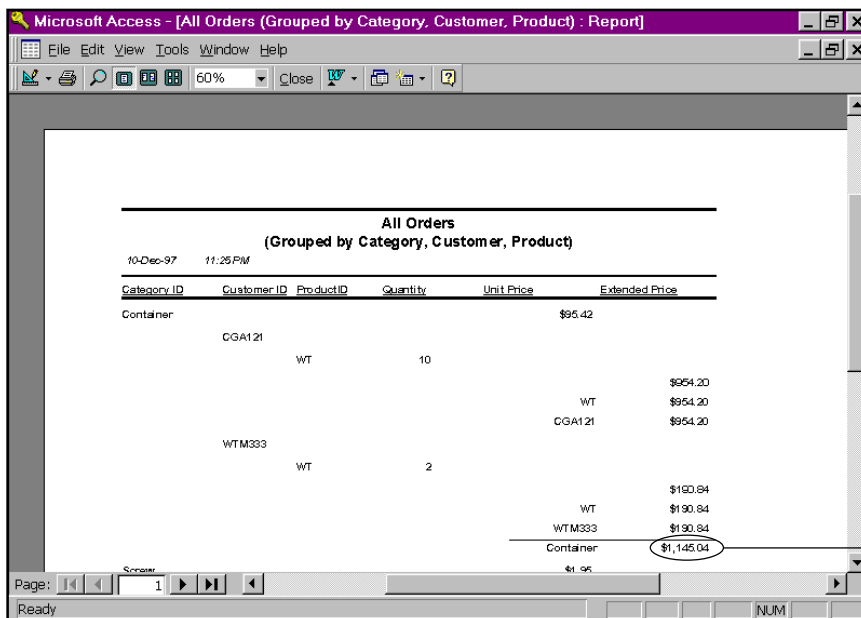
1. Close the All Orders query structure window.
2. Click **Reports tab**
3. View the design of the All Orders report structure.



Mathematical expressions used in report structure

All Orders Report Result

1. Click  (Print Preview)



The All Orders report is based on the All Orders query. All fields, including the mathematical expression, of the query are available to be placed in the report.

Result of mathematical expression in report

Creating A Relational Database

Introduction

In the following exercises you will have the opportunity to create a relational database management system similar to the TEORDER database that you examined in the preceding chapter. This new database, NEWORDER, will have three tables, *Customers*, *Products*, and *Orders*. The tables will be linked to one another in a series of queries, and the queries will become the foundation for printed reports.

Creating The Customers Table

1. Close the current database.
2. Click **File**
3. Click **NewDatabase**
4. Click **Blank Database**
5. Click **OK**
6. Type **NewOrder**
7. Click **Create**

At the table screen:

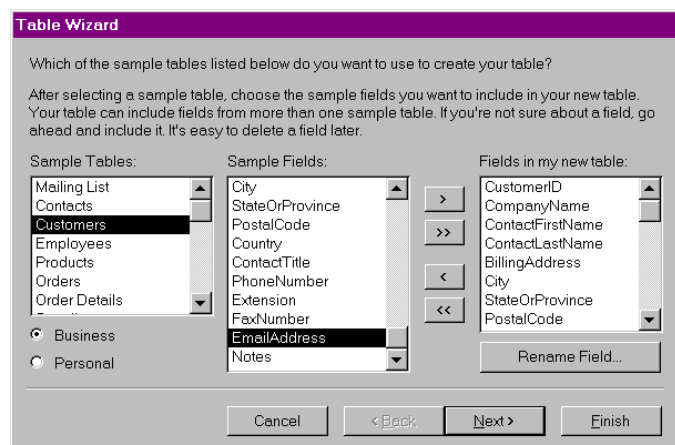
1. Click **New**
2. Click **Table Wizard**
3. Click **OK**
4. Click **Customers Sample Table**

Please add all the listed fields to the new table's structure:

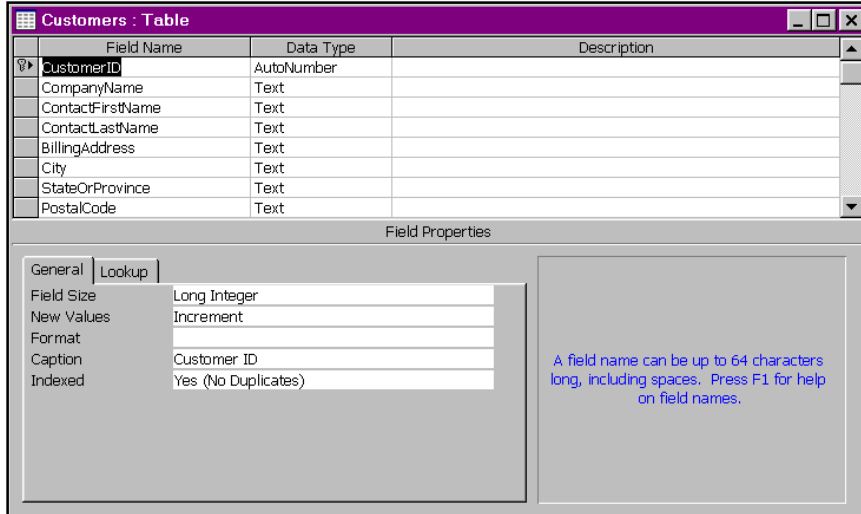
5. Click **>>**

Some of the listed fields are not needed, please remove the following fields from the new table's structure:

CompanyOrDepartment
Country
ContactTitle
Extension
Email Address



6. Click **Next** (2 Times)
7. Click **Modify the table design**
8. Click **Finish**



Modifying The Table Structure

Modify the fields, data types, sizes, and input masks as follows:

Field Name	Data Type	Size	Input Mask
CustomerID	Number	Long Integer	00000
CompanyName	Text	50	
ContactFirstName	Text	15	
ContactLastName	Text	25	
Address	Text	35	
Address2	Text	35	
City	Text	20	
State	Text	2	>LL
PostalCode	Text	14	00000\~9999;0;
PhoneNumber	Text	30	!\(999") "000\~0000;0;
FaxNumber	Text	30	!\(999") "000\~0000;0;
Note	Memo		

Save the table structure:

1. Close the table structure window.
2. Click **Yes**

Input Mask Properties

Here is a list of the input mask characters, excerpted from Microsoft Access' help function:

The InputMask property setting specifies how data is entered and displayed in the text box. For example, if you set this property to 000-00-0000, hyphens are displayed as shown, and an underscore () is displayed in place of each zero.

Note Instead of setting this property to create an input mask, you might find it easier to create an input mask using the Input Mask Wizard. To use this builder, click the Properties button on the toolbar, and then click the Build button in the InputMask property box.

The setting can contain up to three parts separated by semicolons (for example, (999) 000-0000!;0;" " "): The first part specifies the input mask itself (for example, (999) 000-0000!).

The second part specifies whether Microsoft Access stores the literal display characters in the table when you enter data. If you use 0 for this part, all literal display characters (for example, the parentheses in a phone number input mask) are stored with the value; if you enter 1 or leave this part blank, only characters typed into the text box are stored.

The third part specifies the character that Microsoft Access displays for spaces in the input mask. For this part, you can use any character; to display a space, use a space enclosed in quotation marks (" ").

Microsoft Access interprets characters in the first part of the InputMask property setting as shown in the following table.

Character	Description
0	Digit (0-9, entry required, plus [+] and minus [-] signs not allowed).
9	Digit or space (entry not required, plus and minus signs not allowed).
#	Digit or space (entry not required blank positions converted to spaces, plus and minus signs allowed).
L	Letter (A-Z, entry required).
?	Letter (A-Z, entry optional).
A	Letter or digit (entry required).
a	Letter or digit (entry optional).
&	Any character or a space (entry required).
C	Any character or a space (entry optional).
. , : ; - /	Decimal placeholder and thousand, date, and time separators. (The actual character used depends on the settings in the International section of the Microsoft Windows Control Panel).
<	Causes all characters that follow to be converted to lowercase.
>	Causes all characters that follow to be converted to uppercase.
\	Causes the character that follows to be displayed as the literal character (for example, \A is displayed as just A).

Note Setting the InputMask property to the word Password creates a password entry text box. Any character typed in the text box is stored as the character but is displayed as an asterisk (*). You can set this property using the property sheet, a macro, or Access Basic.

When you have defined an input mask and set the Format property for the same data, the Format property takes precedence when the data is displayed. This means that even if you've saved an input mask with data, the input mask is ignored when data is formatted. The data in the underlying table itself isn't changed; the Format property affects only how the data is displayed. When you type data in a field for which you've defined an input mask, the data is always entered in overtyping mode. In addition, if you use the Backspace key to delete a character, the character is deleted, but a blank space remains. If you move text from a field for which you've defined an input mask onto the Clipboard, the literal display characters are copied, regardless of whether you have specified that they be saved with data.

Note Only characters that you type directly in a text box are affected by the input mask. Microsoft Access ignores any input masks when you import data, run an action query, or enter characters in a text box using Access Basic by setting the text box's Text property or in a macro using the SetValue action.

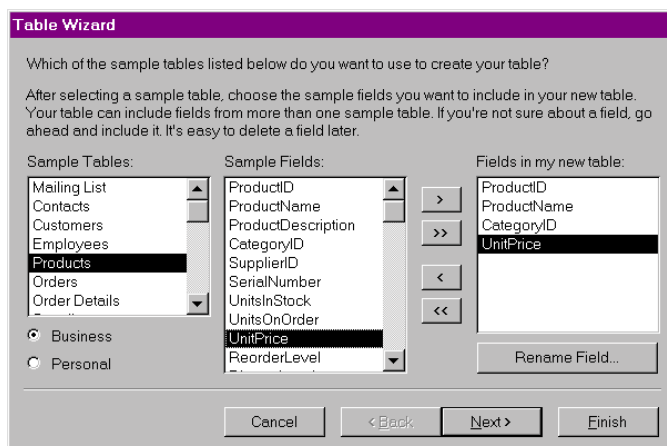
Creating The Products Table

In the table window:

1. Click **New**
2. Click **Table Wizard**
3. Click **OK**
4. Click **Products Sample Table**

Add only four specific fields to the table structure:

1. Click **ProductID**
2. Click **>**
3. Click **ProductName**
4. Click **>**
5. Click **CategoryID**
6. Click **>**
7. Click **UnitPrice**
8. Click **>**



9. Click **Next** (3 Times)
10. Click **Modify the table design**
11. Click **Finish**

Modifying The Table Structure

Modify the fields, data types, and sizes as follows:

Field Name	Data Type	Size
ProductID	Number	Long Integer
ProductName	Text	50
CategoryID	Text	20
UnitPrice	Currency	

Save the table structure:


1. Close the table structure window and save the changes.

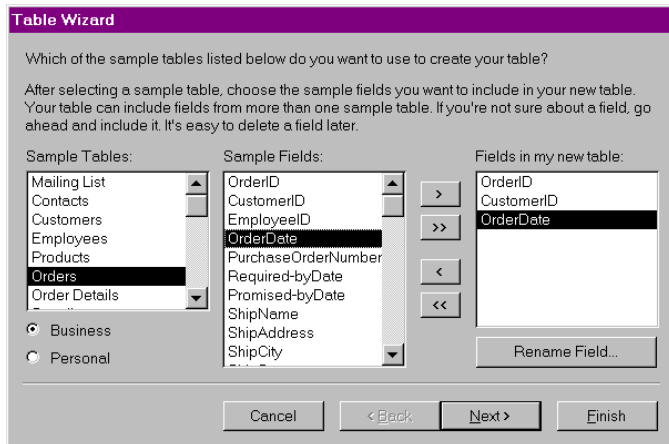
Creating The Orders Table

In the table window:

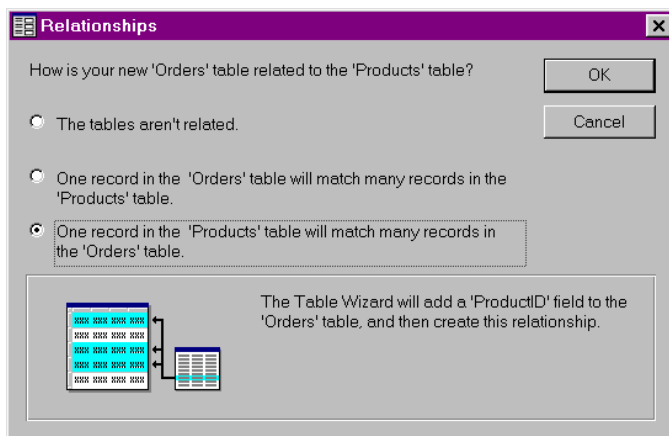
1. Click **New**
2. Click **Table Wizard**
3. Click **OK**
4. Click **Orders Sample Table**

Add only three specific fields to the table structure:

1. Click **OrderID**
2. Click 
3. Click **CustomerID**
4. Click 
5. Click **OrderDate**
6. Click 



7. Click **Next** (2 Times)
8. Click **not related to Products**
9. Click **Relationships...**
10. Click **One record in the 'Products' table will match many records in the 'Orders' table.**



11. Click **OK**
12. Click **Next**
13. Click **Modify the table design**
14. Click **Finish**

Modifying The Table Structure

Modify the fields, data types, and sizes as follows:

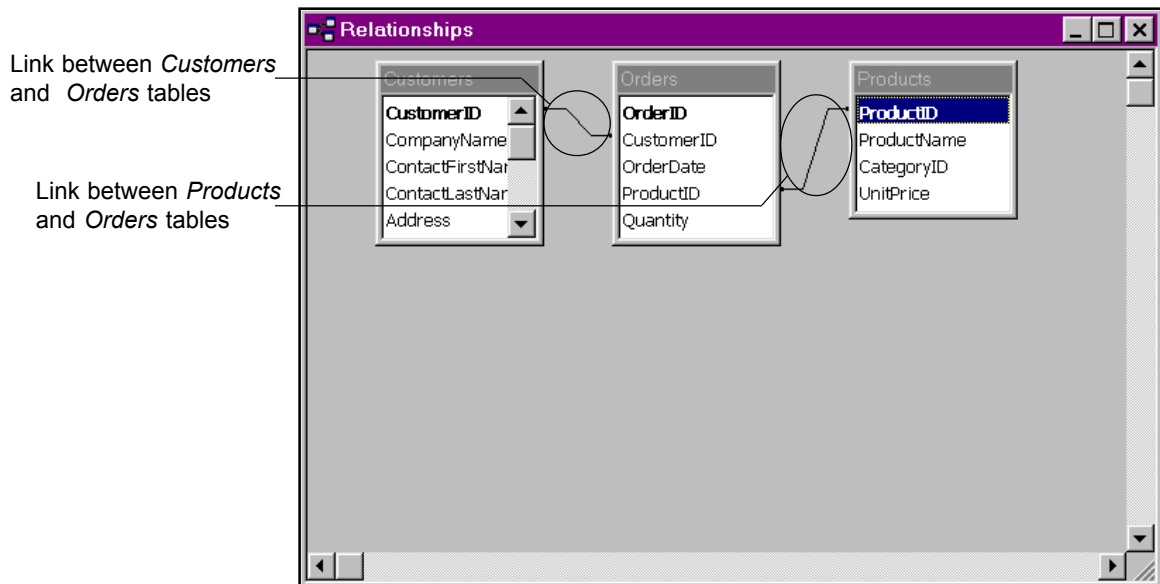
Field Name	Data Type	Size
OrderID	AutoNumber	
CustomerID	Number	Long Integer
OrderDate	Date/Time	
ProductID	Number	Long Integer
Quantity	Number	Long Integer

1. Close the table structure window and save the changes.

Viewing Relationships

While viewing the main database window:

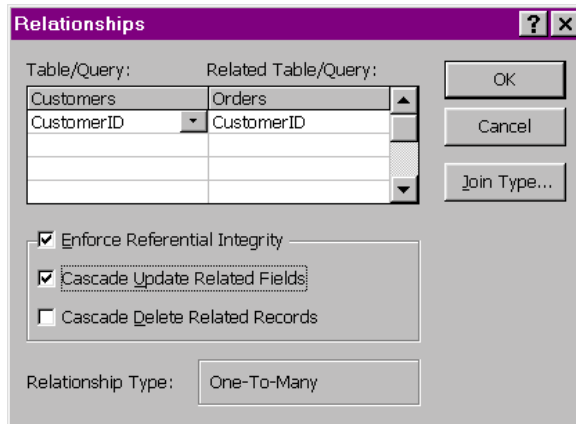
1. Click **Tools**
2. Click **Relationships**



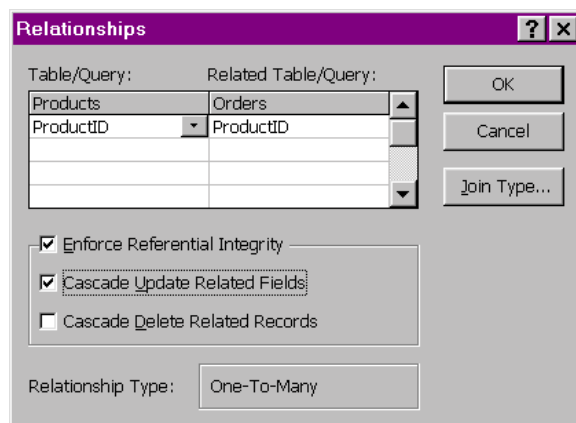
The link between *Customers* and *Orders* as well as the link between *Products* and *Orders* is visible. However, no limits for the relationships have been defined.

Editing Relationships

1. Double-click the line linking the *Customer* and *Orders* tables.
2. Click **Enforce Referential Integrity**
3. Click **Cascade Update Related Fields**



4. Click **OK**
5. Double-click the line linking the *Products* and *Orders* tables.
6. Click **Enforce Referential Integrity**
7. Click **Cascade Update Related Fields**



8. Click **OK**
9. Close the relationships window.

By enforcing referential integrity, new records in the Orders table may be added only for those Customer ID's who have already been entered into the Customer table.

The Cascade Update Related Fields property allows Customer ID data in the Customers table to be edited and the related fields in the Orders table to automatically update the changes.

One-To-Many Relationship


The one-to-many relationships between the *Customers-Orders* and *Products-Orders* tables allows the primary key of the two base tables (*Customers* and *Products*) to be linked to possibly multiple occurrences of the same entry in the *Orders* table.

For example, a given customer, who has one record in the *Customers* table; therefore, has a unique CustomerID, may place multiple orders in the *Orders* table. A single CustomerID, the primary key field of the *Customers* table, links to possibly many sales orders in the *Orders* table.

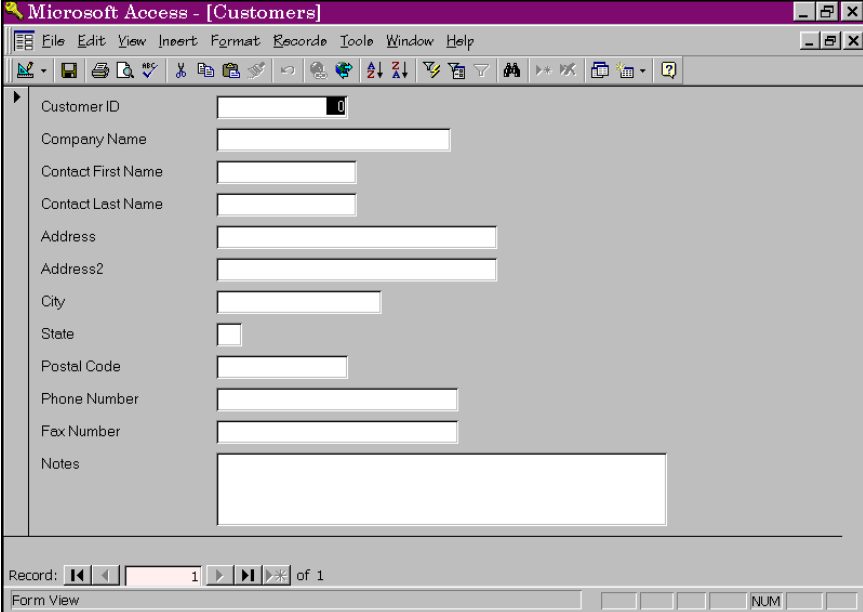
The *Products* table has a similar one-to-many relationship with the *Orders* table. Multiple orders may be placed for the same ProductID. The ProductID is the primary key field of the *Products* table, the base table for the link.

Customers Data Entry

To make data entry easier, create an AutoForm based on the Customers table.

1. Click **Table tab**
2. Click **Customers**
3. Click  (AutoForm)

Your screen should look like this:



The screenshot shows the Microsoft Access interface with the title bar "Microsoft Access - [Customers]". The menu bar includes File, Edit, View, Insert, Format, Records, Tools, Window, and Help. The toolbar contains various icons for file operations and data manipulation. The main area displays an AutoForm for the Customers table with the following fields and their corresponding input controls:

- CustomerID: Text box with a small grid icon on the right.
- Company Name: Text box.
- Contact First Name: Text box.
- Contact Last Name: Text box.
- Address: Text box.
- Address2: Text box.
- City: Text box.
- State: Small text box.
- Postal Code: Text box.
- Phone Number: Text box.
- Fax Number: Text box.
- Notes: Large text area.

At the bottom, the status bar shows "Record: 1 of 1" and "Form View".

Please enter the following records:

<p>CustomerID: 95123 Name: Joe Smith Organization: Waste Management Technologies Address: Old Harvester Road Address2: City, St ZIP: Baltimore, MD 21203-2312 Phone: (410) 392-3231 Fax: (410) 291-4444 Note:</p>	<p>CustomerID: 95126 Name: Dorothy Owsiany Organization: Care Givers of America Address: Rolling Park Road Address2: City, St ZIP: Atholton, MD 21083-2311 Phone: (410) 233-2311 Fax: (410) 233-9449 Note: Elderly care providers</p>
<p>CustomerID: 95124 Name: Billy Washington Organization: ABC Rental Company Address: Suite 100 Address2: 123 Main Street City, St ZIP: Columbia, MD 21044-2424 Phone: (410) 290-0202 Fax: (410) 290-9393 Note: Household and light commercial rentals</p>	<p>CustomerID: 95127 Name: Dave Johnson Organization: Widgets USA Address: Building A Address2: 9891 Landing Road City, St ZIP: Ellicott City, MD 21042-8048 Phone: (410) 992-9323 Fax: (410) 992-9321 Note: Specialty connectors for electrical applications</p>
<p>CustomerID: 95125 Name: Paul Rogers Organization: Baby Supplies, Inc Address: 9 West Avenue Address2: City, St ZIP: Baltimore, MD 21202-9321 Phone: (410) 321-3423 Fax: (301) 291-2113 Note:</p>	

4. When data entry is complete, close and save the form.

Products Data Entry

1. Create a data entry form based on the Products table. When finished, close and save the form.

Please enter the following records:

<p>ProductID: 87546 ProductName: Pail, 5 gallon CategoryID: Container UnitPrice: 5.95</p>	<p>ProductID: 95456 ProductName: Screw, Crossthreaded CategoryID: Screw UnitPrice: .05</p>	<p>ProductID: 89456 ProductName: Basket, Laundry CategoryID: Container UnitPrice: 7.95</p>
<p>ProductID: 12545 ProductName: Widget, Blue CategoryID: Widget UnitPrice: 18.75</p>	<p>ProductID: 78156 ProductName: Screw, Inverted CategoryID: Screw UnitPrice: .07</p>	
<p>ProductID: 45987 ProductName: Waste Container, 33 gallon CategoryID: Container UnitPrice: 17.50</p>	<p>ProductID: 12355 ProductName: Widget, Green CategoryID: Widget UnitPrice: 19.95</p>	

Orders Data Entry

1. Create a data entry form based on the Orders table. When finished, close and save the form.

Please enter the following records:

CustomerID: 95123 OrderDate: 1/1/96 ProductID: 12355 Quantity: 5	CustomerID: 95126 OrderDate: 1/8/96 ProductID: 89456 Quantity: 4
CustomerID: 95123 OrderDate: 1/2/96 ProductID: 12545 Quantity: 4	CustomerID: 95126 OrderDate: 1/9/96 ProductID: 78156 Quantity: 5
CustomerID: 95124 OrderDate: 1/5/96 ProductID: 12355 Quantity: 5	CustomerID: 95127 OrderDate: 1/10/96 ProductID: 87546 Quantity: 4
CustomerID: 95124 OrderDate: 1/5/96 ProductID: 45987 Quantity: 1	CustomerID: 95127 OrderDate: 1/11/96 ProductID: 89456 Quantity: 11
CustomerID: 95125 OrderDate: 1/6/96 ProductID: 12355 Quantity: 5	CustomerID: 95127 OrderDate: 1/12/96 ProductID: 78156 Quantity: 2
CustomerID: 95125 OrderDate: 1/7/96 ProductID: 12545 Quantity: 20	

Form Navigation

Introduction

In relational databases, you may need to view information on another topic quickly and easily. Creating command buttons on your forms allow you to open and close other forms in the database.

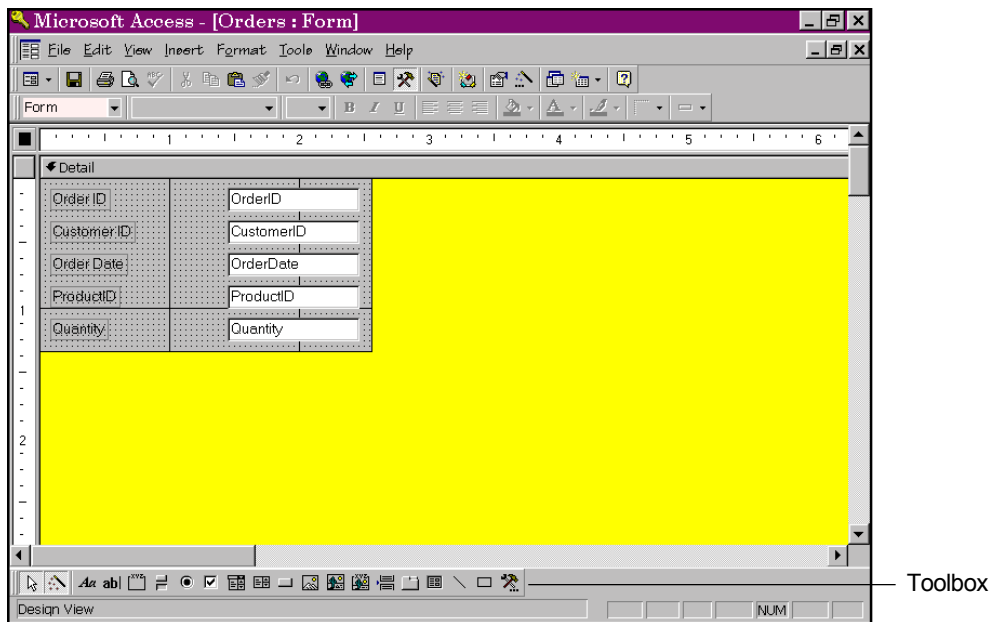
Creating a Command Button


Command buttons can perform a wide variety of operations, but we will focus on Form Navigation. For the following exercises, make sure the *Toolbox* is visible.

1. View the design of the **Orders** form.

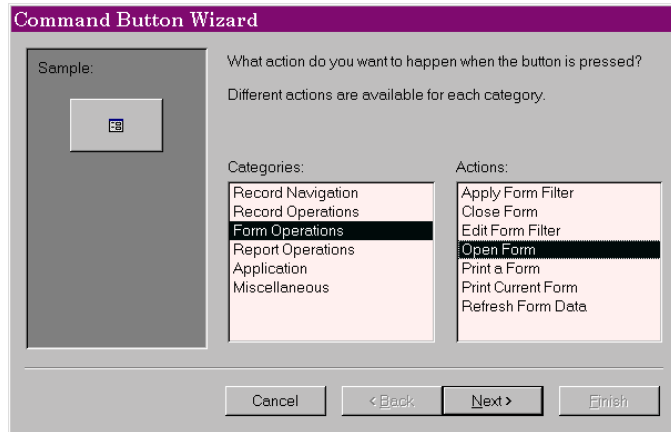
Your screen should look like this:

If the Toolbox is not visible, click View / Toolbox from the menu.

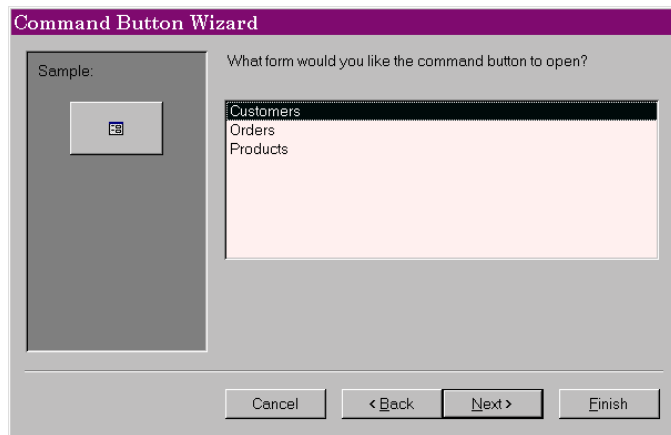


2. Click  (**Command Button**)
3. Click anywhere on the form detail. The Command Button Wizard will launch.

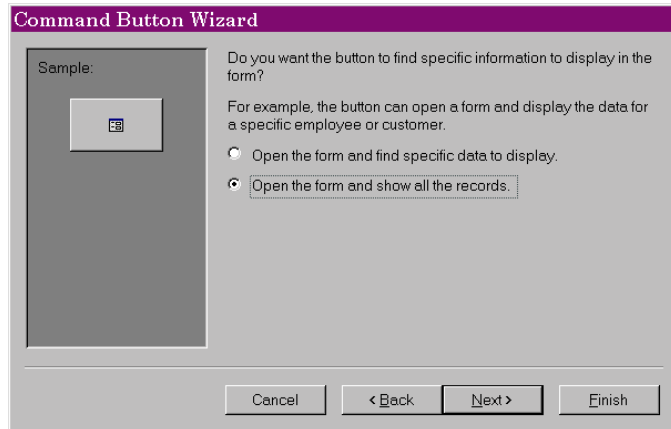
4. Click **Form Operations**
5. Click **Open Form**



6. Click **Next >**
7. Click **Customers**



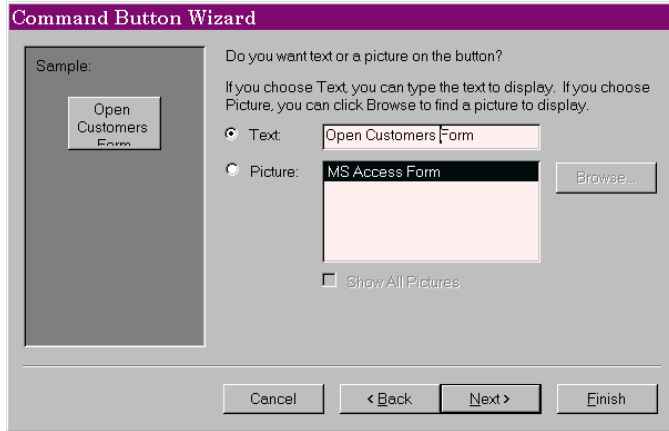
8. Click **Next >**
9. Type **Open the form and show all the records**



10. Click **Next >**

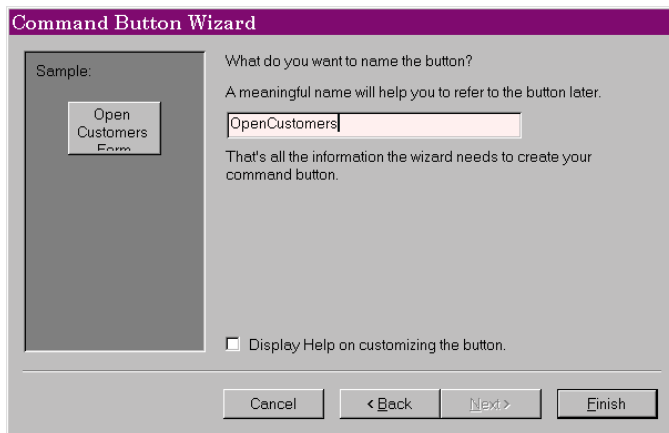
11. Click **Text**

12. Type **Open Customers Form**

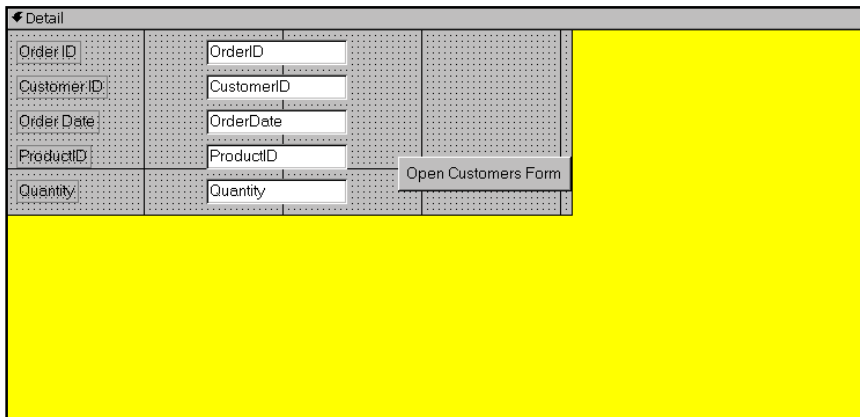


13. Click **Next >**

14. Type **OpenCustomers**



15. Click **Finish**



You should view the Form View and check to see that your new command button works. It should open the Customers Form.

Exercise

Create the following command buttons on these forms:

Form	Function of the Command Button
Orders	Open the Products Form
Orders	Close the Orders Form
Customers	Close the Customers Form
Products	Close the Products Form

Relational Queries

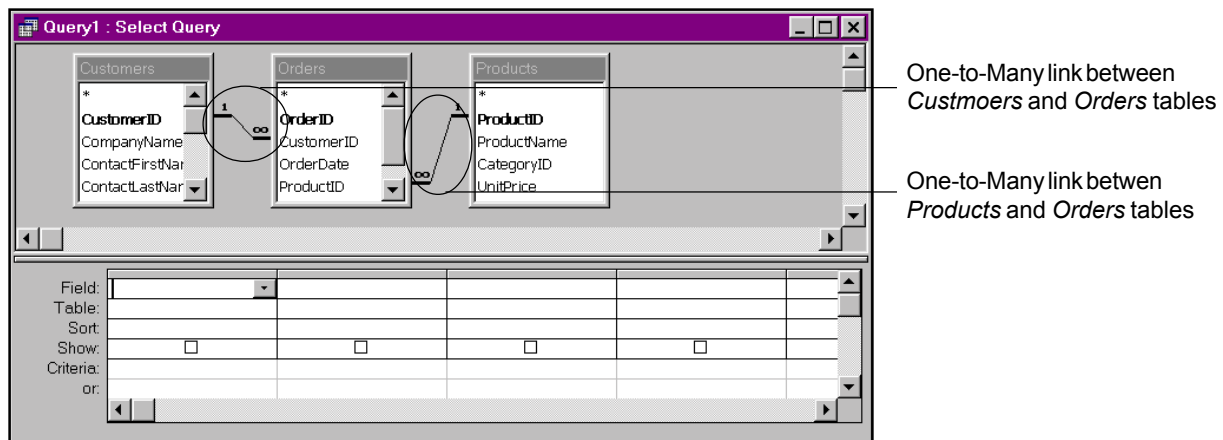
Introduction

In this exercise you will create a relational query of the *Customers*, *Products*, and *Orders* tables.

By relating “linking” the three tables, they may be used as a cohesive database system. As a database, mathematical expressions may be evaluated using fields from more than one table.

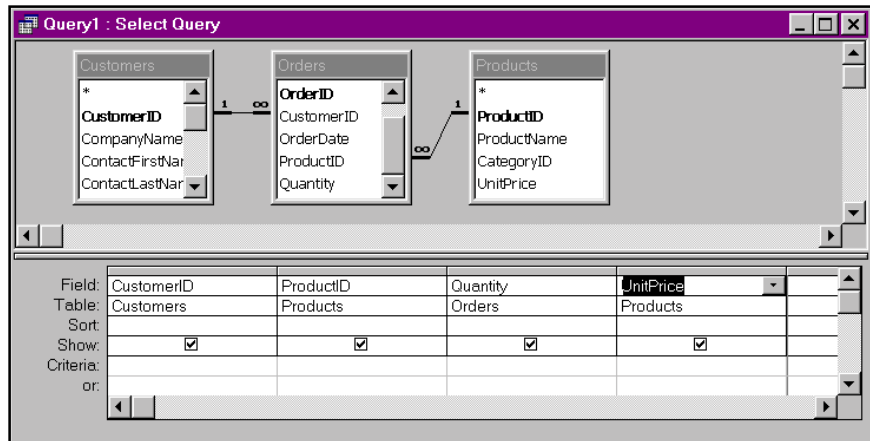
Creating A Relational Query

1. Click **Queries** tab
2. Click **New**
3. Click **Design View**
4. Click **OK**
5. Double-click *Customers*
6. Double-click *Orders*
7. Double-click *Products*
8. Click **Close**



Adding Fields To The Query View

1. Double-click **CustomerID** (in the Customers Table)
2. Double-click **ProductID** (in the Products Table)
3. Double-click **Quantity**
4. Double-click **UnitPrice**



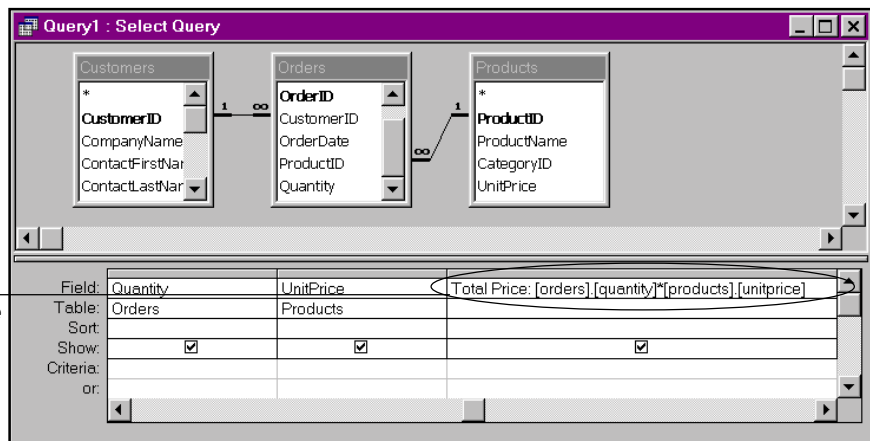
Adding A Mathematical Expression Field

In the field to the right of UnitPrice:

1. Type: **Total Price:orders.quantity*products.unitprice**
2. Press **<Enter>**

When selecting a field from a table in an expression, enter the table name.field name (table name, period, field name).

Mathematical Expression evaluated the *Quantity x UnitPrice*



Running The Query

Run the query:

1. Click  (Run)

To return to the Design View window:

2. Click  (Design View)

Saving The Query

1. Close the query window
2. Save it as *Orders - Quantity*UnitPrice*

Relational Reports

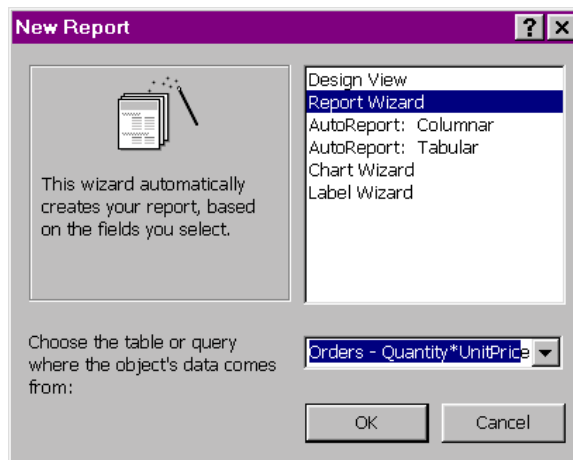
Introduction

In the following exercises you will create a report that uses fields from all three tables: *Customers*, *Products*, and *Orders*. The report will be based on the relational query *Orders - Quantity*UnitPrice*.

Creating A Relational Report

From the main database window:

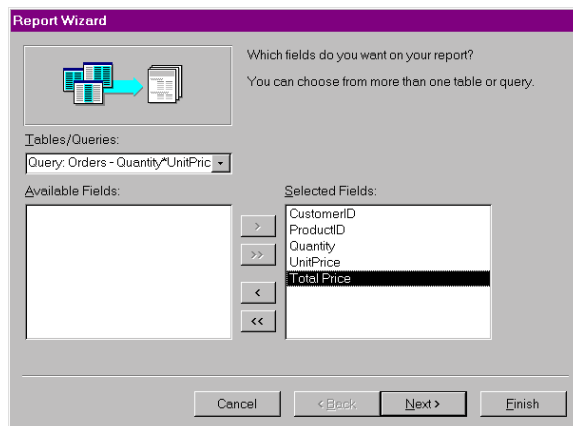
1. Click **Reports tab**
2. Click **New**
3. Click **Report Wizards**
4. Select the query *Orders - Quantity*Unit Price*.



5. Click **OK**

Add all available fields to the report structure:

1. Click **>>**

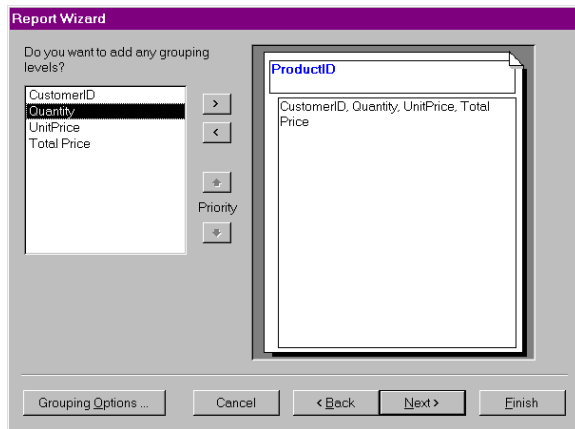


2. Click **Next**

Group the report by the ProductID field:

1. Click **ProductID**

2. Click **>**

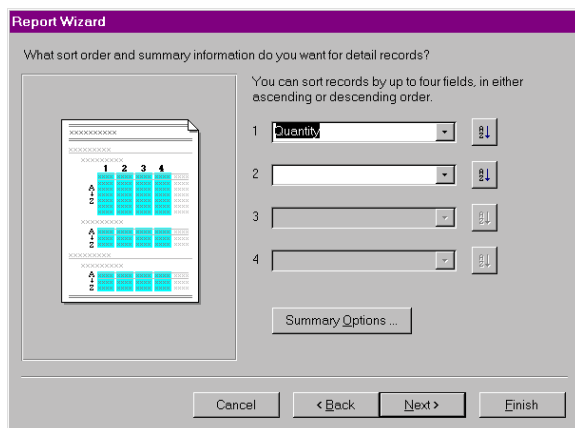


3. Click **Next**

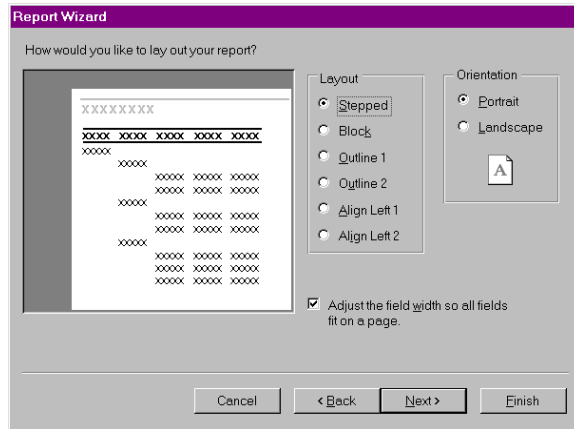
Sort the groupings by Quantity. This will sort the quantities of products purchased within each grouping:

1. Click **Quantity**

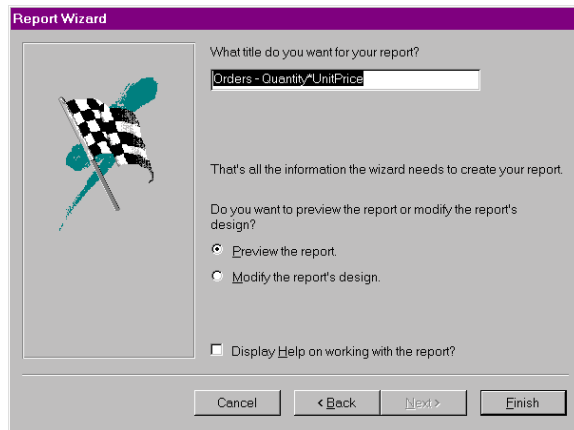
2. Click **>**



3. Click **Next**



4. Click **Next** (2 times)



View The Finished Report

1. Click **Finish**

The screenshot shows a Microsoft Access window titled "Microsoft Access - [Orders - Quantity*UnitPrice]". The report is displayed in a preview window. The report title is "Orders - Quantity*UnitPrice". The data is presented in a table with the following columns: ProductID, Quantity, Customer ID, Unit Price, and Total Price. The data is grouped by ProductID.

ProductID	Quantity	Customer ID	Unit Price	Total Price
12355	5	05125	\$19.95	\$99.75
	5	05124	\$19.95	\$99.75
	5	05123	\$19.95	\$99.75
12545	4	05123	\$18.75	\$75.00
	20	05125	\$18.75	\$375.00
45987	1	05124	\$17.00	\$17.00
78156	2	05127	\$0.07	\$0.14
	5	05126	\$0.07	\$0.35

The report is displayed in a preview window with a status bar at the bottom showing "Page: 1" and "Ready".

Saving The Report

1. Close the report window

The report will be automatically saved and assigned the report title as the report name.

Index

C

Cascade 17
Create 10
create 1, 3, 10, 12, 25, 28

D

Database 3, 10
database 1, 2, 3, 4, 10, 16, 25

E

Expression 26
expression 4, 25, 26

F

Field 1, 2, 4
field 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 13, 14, 16, 18

G

grouping 29

I

Input Mask 6, 12
input mask 6, 7, 11, 12, 13

L

Linking 2
linking 17, 25, 34, 35

O

One-To-Many 18
One-to-Many 25
one-to-many 18

P

Print Preview 9
property 12

Q

Query 8, 25, 26, 27
query 8, 9, 25, 27

R

Record 2
record 2, 3
Referential Integrity 17
referential integrity 17
Relational 3, 5, 10, 25, 28
relational 3, 5, 10, 25, 28
Relations 4
Relationship 16, 17, 18
relationship 8, 16, 17, 18
Report 9, 28, 31
report 9, 28, 29, 31
Report Wizard 28
Run 8, 27

S

Sort 29
Structure 1, 2, 4
structure 1, 2, 3

T

Table 3, 4, 6, 7, 10, 11, 13, 14, 16
table 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18
Table Structure 6, 7, 11, 14, 16
table structure 7, 8, 11, 13, 14, 16
Table Wizard 10, 13, 14