



Queries



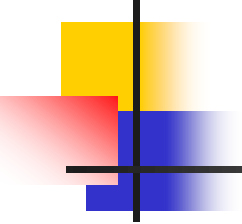
What is a query?

- A query is actually the name for any database manipulation operation.
- The most commonly used type is a *select query*, which is a way of selecting what fields in what records are to be displayed.



Types of queries you can create in Microsoft Access

- Select queries
- Crosstab queries (actually a special kind of select query)
- Action queries
 - Make-table queries
 - Delete queries
 - Update queries
 - Append queries)

- 
-
- In addition, any of these types can be designed to prompt the user for information, in which case it is called a *parameter* query.

Queries: What they are and how they work

You use queries to view, change, and analyze data in different ways. You can also use them as the source of records for forms and reports.

Bring together data from multiple tables and sort it in a particular order.

Products and Suppliers : Select Query

Product Name	Supplier	Phone
Alice Mutton	Pavlova, Ltd.	(03) 44
Aniseed Syrup	Exotic Liquids	(71) 55
Camembert Pierrot	Gai pâturage	38.7
Carnarvon Tigers	Pavlova, Ltd.	(03) 44
Chai	Exotic Liquids	

Perform calculations on groups of records.

Sales Totals : Select Query

Name	Total Orders	Total Sales
Andrew Fuller	125	\$197,110.84
Anne Dodsworth	53	\$86,737.34

Calculate a sum, count, or another type of total, and then group the results by two types of information — one down the left side of the datasheet and another across the top.

Quarterly Orders by Product : Crosstab Query

Category Name	Qtr 1	Qtr 2	Qtr 3	Q
Beverages	\$7,696.11	\$26,942.16	\$13,800.18	\$3
Condiments	\$2,666.22	\$5,449.50	\$5,988.40	\$1
Confections	\$7,737.14	\$6,175.75	\$17,118.93	\$1
Dairy Products	\$13,665.87	\$10,494.94	\$15,921.14	\$2
Grains/Cereals	\$11,624.00	\$9,160.84	\$3,685.07	\$1

Queries: What they are and how they work

The most common type of query is a select query. A select query retrieves data from one or more tables using criteria you specify, and then displays it in the order you want.

When you run the query, Microsoft Access retrieves the records you specify...

... and then displays the combined data in the order you want.

Products : Table	
Product Name	Units
Alice Mutton	
Aniseed Syrup	
Boston Crab Meat	
Camembert Pierrot	

Suppliers : Table		
Supplier	Supplier ID	Phone
Exotic Liquids	1	(71) 555-2222
Forêts d'érables	29	(514) 555-2955
Formaggi Fortini s.r.l.	14	(0544) 60323
G'day Mate	24	(02) 555-5914
Pavlova, Ltd.	62	(03) 444-2343
		(06) 431-7877
		503) 555-9931
		(03) 444-2343

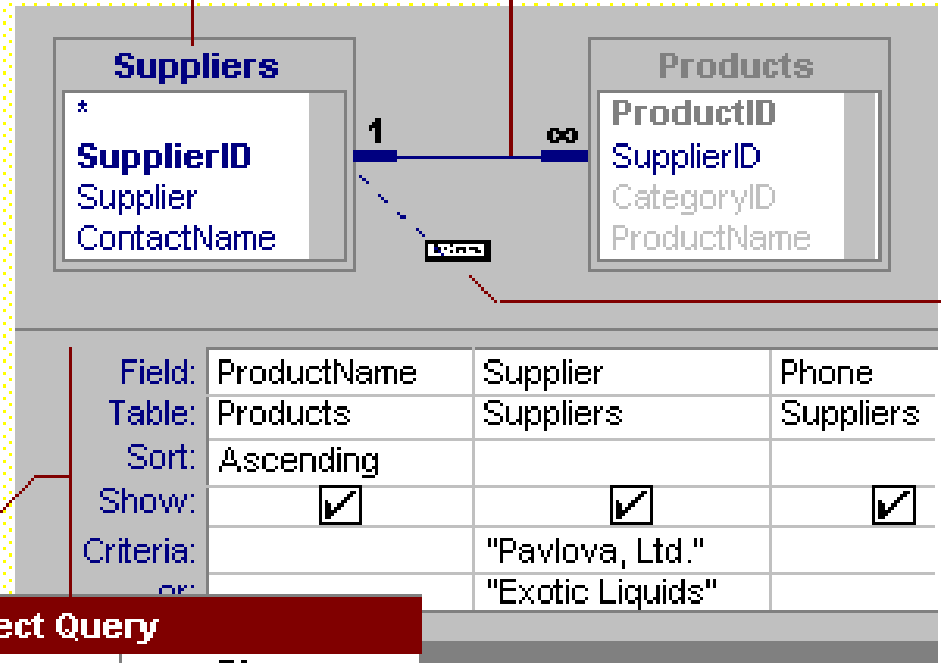
Products and Suppliers : Select Query		
Product Name	Supplier	Phone
Alice Mutton	Pavlova, Ltd.	(03) 444-2343
Aniseed Syrup	Exotic Liquids	(71) 555-2222
Boston Crab Meat	New England Seafood	(617) 555-3267
Camembert Pierrot	Gai pâturage	38.76.98.06

Queries: What they are and how they work

You create a query with a wizard or from scratch in query Design view. In Design view, you specify the data you want to work with by adding the tables or queries that contain the data, and then by filling in the design grid.

Field lists show the fields in the tables or queries you add to your query.

A join line tells Microsoft Access how data in one table or query is related to data in another table or query.



The fields, sort order, and criteria you add to the design grid determine what you will see in the query's results.

You add fields to the design grid by dragging them from the field lists.

Products and Suppliers : Select Query

Product Name	Supplier	Phone
Alice Mutton	Pavlova, Ltd.	(03) 444-2343
Aniseed Syrup	Exotic Liquids	(71) 555-2222
Carnarvon Tigers	Pavlova, Ltd.	(03) 444-2343



Creating a query

- You can create only very simple queries by using the Query Wizard.
- We'll create all our queries using the Design View for queries.
- Once designed, the results of a query can be displayed in Datasheet View.



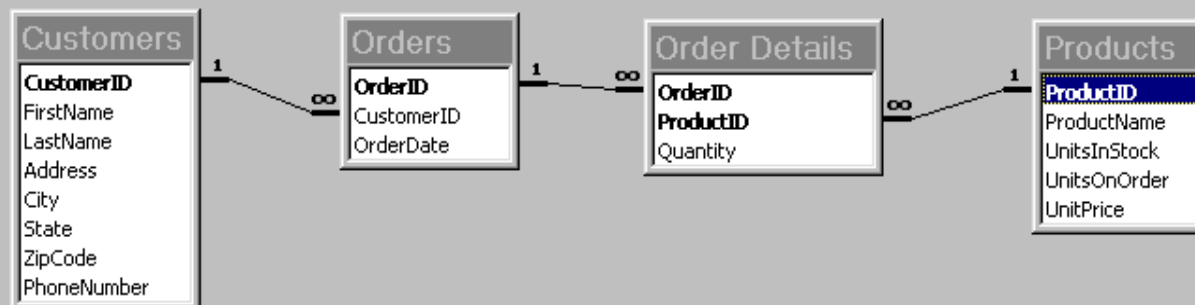
Select queries

- A select query is the most common type of query.
- It retrieves data from one or more tables and displays the results in a datasheet where you can update the records (with some restrictions).
- You can also use a select query to group records and calculate sums, counts, averages, and other types of totals.



Computer store example

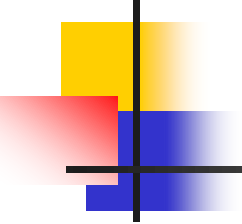
- Let's consider the following database as an example. (More precisely, this is a *database schema*, or database layout, independent of the actual data it contains.)
- It consists of 4 related tables, containing fields as indicated.

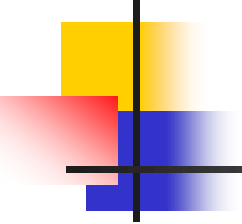


- 
-
- Let's suppose the Customers table contains the following data:



	CustomerID	First Name	Last Name	Address	City	State	Zip Code	Phone Number
▶ +	C0001	Benjamin	Lee	1000 Call Street	Tallahassee	FL	33340	(904) 327-4124
+	C0002	Eleanor	Milgrom	7245 NW 8 Street	Margate	FL	33065	(305) 974-1234
+	C0003	Neil	Goodman	4215 South 81 Street	Margate	FL	33065	(305) 444-5555
+	C0004	Nicholas	Colon	9020 N.W. 75 Street	Coral Springs	FL	33065	(305) 753-9887
+	C0005	Michael	Ware	276 Brickell Avenue	Miami	FL	33131	(305) 444-3980
+	C0006	Jeffrey	Muddell	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
+	C0007	Ashley	Geoghegan	7500 Center Lane	Coral Springs	FL	33070	(305) 753-7830
+	C0008	Serena	Sherard	5000 Jefferson Lane	Gainesville	FL	32601	(904) 375-6442
+	C0009	Luis	Couto	455 Bargello Avenue	Coral Gables	FL	33146	(305) 666-4801
+	C0010	Derek	Anderson	6000 Tigertail Avenue	Coconut Grove	FL	33120	(305) 446-8900
+	C0011	Lauren	Center	12380 S.W. 137 Avenue	Miami	FL	33186	(305) 385-4432
+	C0012	Robert	Slane	4508 N.W. 7 Street	Miami	FL	33131	(305) 635-3454
*	(AutoNumber)							

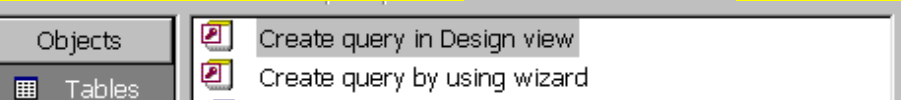
- 
-
- Let's create a select query that shows only those records for customers in Miami.
 - We double-click "Create query in Design view" in the Database Window.
 - "Select Query" will be the default query type when we start in Design View, but we can go to the Query Type button to set it as desired.

- 
-
- We then click the Show Table button to select the tables the query is to use. In this case we use only the Customers table.
 - The query design window has an upper pane and a lower pane.
 - The upper pane shows the tables we've selected and their relationships, while the lower pane shows the *design grid*.



Query Type button

Show Table button



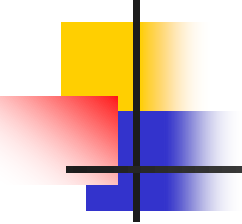
Query1 : Select Query

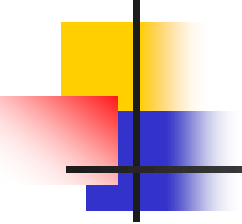
Customers

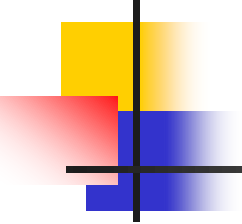
- *
 - CustomerID
 - FirstName
 - LastName
 - Address
 - City
 - State
 - ZipCode
 - PhoneNumber

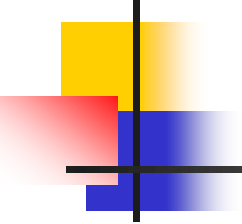
Field:	CustomerID	LastName	FirstName	Address	City	State	ZipCode	PhoneNumber	
Table:	Customers	Customers	Customers	Customers	Customers	Customers	Customers	Customers	
Sort:	Ascending								
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:					Miami				
or:									

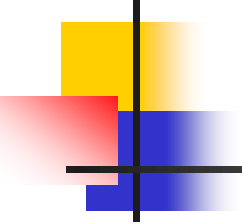
Design grid

- 
-
- Initially, the design grid is empty, but we use it to specify the data we want the query to retrieve.
 - Each column of the design grid represents something we want the query to display (if its check box is so marked).

- 
-
- In this case we wanted all the fields of the table to be displayed, so we copied the field names from the upper pane to the lower pane.
 - There are several ways to do this.
 - One is to use the drop-down lists that are available at the top of each column in the design grid. This is too slow when there are more than 3 or 4 columns.

- 
-
- Another way is to click and drag each field name in the upper pane to a column in the design grid, but this is also too slow.
 - Still another way is to double-click each field name in the upper pane.

- 
-
- But the fastest way when there are many fields is to Ctrl+click each desired field from the upper pane and then drag the entire set of fields to the design grid.
 - This was what was done in this example, but note that we also changed the ordering, putting Last Name before First Name. For this, we just used the drop-down lists to make these minor changes.

- 
-
- In addition to using the design grid to specify which fields to display, we have also specified that:
 - the records this query returns are to be sorted in ascending order of Customer ID; and
 - only those records whose City field matches 'Miami' are to be returned.



Computer Store Example : Database

Open Design New

Objects

Create query in Design view

Create query by using wizard

Query1 : Select Query

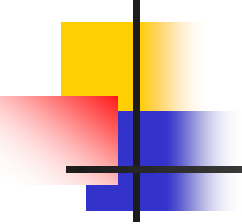
Customers

*
CustomerID
FirstName
LastName
Address
City
State
ZipCode
PhoneNumber

Field:	CustomerID	LastName	FirstName	Address	City	State	ZipCode	PhoneNumber	
Table:	Customers	Customers	Customers	Customers	Customers	Customers	Customers	Customers	
Sort:	Ascending								
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:					Miami				
or:									

Sort in ascending
order of customer id

City must
match "Miami"

- 
-
- If we switch to Datasheet View for this query, we get this, the result of running the query on our database:



Computer Store Example : Database

Open Design New

Objects

- Create query in Design view
- Create query by using wizard

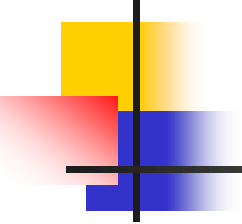
Tables

Query1 : Select Query

	CustomerID	Last Name	First Name	Address	City	State	Zip Code	Phone Number
▶	C0005	Ware	Michael	276 Brickell Avenue	Miami	FL	33131	(305) 444-3980
	C0006	Muddell	Jeffrey	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
	C0011	Center	Lauren	12380 S.W. 137 Avenue	Miami	FL	33186	(305) 385-4432
	C0012	Slane	Robert	4508 N.W. 7 Street	Miami	FL	33131	(305) 635-3454
*	(AutoNumber)							

Record: 1 of 4

Order...

- 
-
- If we want the same data displayed in alphabetical order according to last name, then first name, we'd modify the query's design to look like this:



Computer Store Example : Database

Open Design New

Objects

Tables

- Create query in Design view
- Create query by using wizard

Query1 : Select Query

- Customers
- *
 - CustomerID
 - FirstName
 - LastName
 - Address
 - City
 - State
 - ZipCode
 - PhoneNumber

Field:	CustomerID	LastName	FirstName	Address	City	State	ZipCode	PhoneNumber	
Table:	Customers	Customers	Customers	Customers	Customers	Customers	Customers	Customers	
Sort:		Ascending	Ascending						
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Criteria:					"Miami"				
or:									

- 
-
- The result of running this query is:



Computer Store Example : Database

Open Design New

Objects

Tables

- Create query in Design view
- Create query by using wizard

Query1 : Select Query

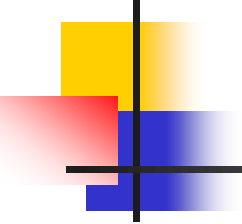
	CustomerID	Last Name	First Name	Address	City	State	Zip Code	Phone Number
▶	00011	Center	Lauren	12380 S.W. 137 Avenue	Miami	FL	33186	(305) 385-4432
	C0006	Muddell	Jeffrey	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
	C0012	Slane	Robert	4508 N.W. 7 Street	Miami	FL	33131	(305) 635-3454
	C0005	Ware	Michael	276 Brickell Avenue	Miami	FL	33131	(305) 444-3980
*	(AutoNumber)							

Record: 1 of 4

- 
-
- Now let's suppose the Orders table contains the following data:



	OrderID	Customer ID	Order Date
▶ +	00001	C0004	4/15/99
+	00002	C0003	4/18/99
+	00003	C0006	4/18/99
+	00004	C0007	4/18/99
+	00005	C0001	4/20/99
+	00006	C0001	4/21/99
+	00007	C0002	4/21/99
+	00008	C0002	4/22/99
+	00009	C0001	4/22/99
+	00010	C0002	4/22/99
+	00011	C0001	4/24/99
+	00012	C0007	4/24/99
+	00013	C0004	4/24/99
+	00014	C0006	4/25/99
+	00015	C0009	4/25/99
+	00016	C0006	4/26/99
+	00017	C0011	4/26/99
+	00018	C0011	4/26/99
+	00019	C0012	4/27/99
+	00020	C0012	4/28/99
+	00021	C0010	4/29/99
+	00022	C0010	4/29/99
+	00023	C0008	4/30/99
+	00024	C0006	5/1/99
+	00025	C0006	5/1/99
*	(AutoNumber)		10/28/00

- 
-
- Let's create a query that displays all orders whose date is prior to 4/20/99 or on or after 5/1/99.
 - Here's the Design View of this query:



Computer Store Example : Database

Query2 : Select Query

Orders

- *
 - OrderID
 - CustomerID
 - OrderDate

Field:	OrderID	CustomerID	OrderDate				
Table:	Orders	Orders	Orders				
Sort:							
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Criteria:			>=#5/1/99#				
or:			<=#4/20/99#				

Order... [Maximize] [Close]

- 
-
- And here's what this query returns:



Computer Store Example : Database

Query2 : Select Query

	OrderID	Customer ID	Order Date
▶	00001	C0004	4/15/99
	00002	C0003	4/18/99
	00003	C0006	4/18/99
	00004	C0007	4/18/99
	00024	C0006	5/1/99
	00025	C0006	5/1/99
*	(AutoNumber)		10/29/00

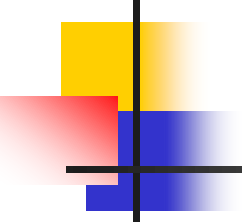
Record: 1 of 6

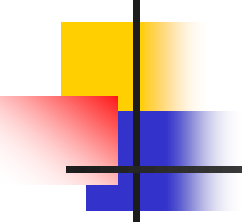
Order...



Criteria in the design grid

- The cells in the design grid labeled Criteria: and or:, and all those below them, are interpreted as follows:

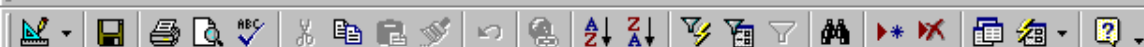
- 
-
- If the expressions are in different cells in the same row, Microsoft Access uses the **And** operator, which means only the records that meet the criteria in all the cells will be returned.

- 
-
- If the expressions are in different rows of the design grid, Microsoft Access uses the **Or** operator, which means records that meet criteria in any of the cells will be returned.

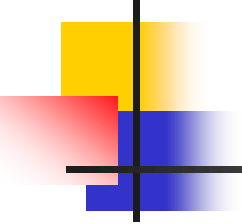


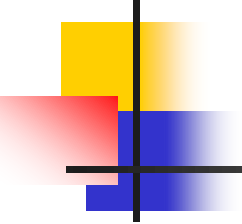
Adding a calculated field

- Here is the Products table from our database:



	ProductID	Product Name	Units In Stock	Units On Order	UnitPrice
▶	P0001	Pentium II/300 MHz	50	0	\$1,899.00
	P0002	Pentium II/350 MHz	25	5	\$1,999.00
	P0003	Pentium II/400 MHz	125	15	\$2,099.00
	P0004	Pentium II/450 MHz	25	50	\$2,299.00
	P0005	Pentium II laptop/266MHz	15	25	\$2,599.00
	P0006	15" SVGA Monitor	50	0	\$499.00
	P0007	17" SVGA Monitor	25	10	\$899.00
	P0008	20" Multisync Monitor	50	20	\$1,599.00
	P0009	2.5 GB IDE Hard Drive	15	20	\$399.00
	P0010	2 GB SCSI Hard Drive	25	15	\$799.00
	P0011	4 GB SCSI Hard Drive	10	0	\$1,245.00
	P0012	CD-ROM: 32X	40	0	\$249.00
	P0013	DVD	50	15	\$449.95
	P0014	HD Floppy Disks	500	200	\$9.99
	P0015	Zip Cartridges	100	50	\$14.79
	P0016	Internal Zip Drive	15	3	\$179.95
	P0017	Serial Mouse	150	50	\$69.95
	P0018	Trackball	55	0	\$59.95
	P0019	Joystick	250	100	\$39.95
	P0020	Fax/Modem 56 Kbps	35	10	\$189.95
	P0021	Fax/Modem 33.6 Kbps	20	0	\$65.95
	P0022	Laser Printer (network)	100	15	\$1,395.00
	P0023	Ink Jet Printer	50	50	\$249.95
	P0024	Laser Printer (personal)	125	25	\$569.95
	P0025	Windows 95	400	200	\$95.95
	P0026	Windows 98	150	50	\$75.95
	P0027	Norton Anti-Virus	150	50	\$115.95
	P0028	Microsoft Scenes Screen Saver	75	25	\$29.95
	P0029	Microsoft Bookshelf	250	100	\$129.95
	P0030	Microsoft Cinemania	25	10	\$59.95
	P0031	Professional Photos on CD-ROM	15	0	\$45.95
*	(AutoNumber)				

- 
-
- Let's create a query that computes for every record in this table the total retail value of our inventory of that product.
 - That is, the query should compute $[\text{UnitsInStock}] * [\text{UnitPrice}]$ for each product.

- 
-
- Furthermore, let's have the query return only those records where this total retail value is at least \$10,000.
 - Here's our query design:

Microsoft Access - [Query1 : Select Query]

File Edit View Insert Query Tools Window Help

Grid Save Print Undo Cut Copy Paste Undo Redo Refresh All Print Home Refresh Refresh Help

Products

*

ProductID

ProductName

UnitsInStock

UnitsOnOrder

UnitPrice

Field:	ProductID	ProductName	UnitsInStock	UnitsOnOrder	UnitPrice	TotalValue: [UnitsInStock]*[UnitPrice]		
Table:	Products	Products	Products	Products	Products			
Sort:								
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Criteria:						>10000		
or:								

Ready

- 
-
- And here's what this query returns:

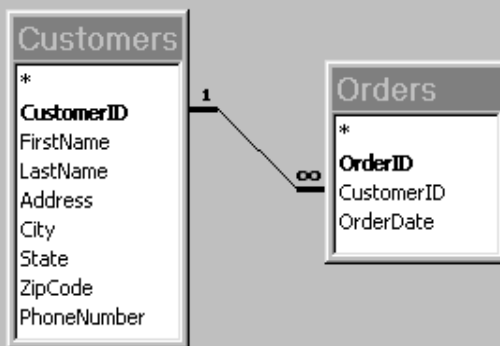
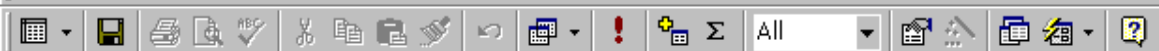


ProductID	Product Name	Units In Stock	Units On Order	UnitPrice	TotalValue
P0001	Pentium II/300 MHz	50	0	\$1,899.00	\$94,950.00
P0002	Pentium II/350 MHz	25	5	\$1,999.00	\$49,975.00
P0003	Pentium II/400 MHz	125	15	\$2,099.00	\$262,375.00
P0004	Pentium II/450 MHz	25	50	\$2,299.00	\$57,475.00
P0005	Pentium II laptop/266MHz	15	25	\$2,599.00	\$38,985.00
P0006	15" SVGA Monitor	50	0	\$499.00	\$24,950.00
P0007	17" SVGA Monitor	25	10	\$899.00	\$22,475.00
P0008	20" Multisync Monitor	50	20	\$1,599.00	\$79,950.00
P0010	2 GB SCSI Hard Drive	25	15	\$799.00	\$19,975.00
P0011	4 GB SCSI Hard Drive	10	0	\$1,245.00	\$12,450.00
P0013	DVD	50	15	\$449.95	\$22,497.50
P0017	Serial Mouse	150	50	\$69.95	\$10,492.50
P0022	Laser Printer (network)	100	15	\$1,395.00	\$139,500.00
P0023	Ink Jet Printer	50	50	\$249.95	\$12,497.50
P0024	Laser Printer (personal)	125	25	\$569.95	\$71,243.75
P0025	Windows 95	400	200	\$95.95	\$38,380.00
P0026	Windows 98	150	50	\$75.95	\$11,392.50
P0027	Norton Anti-Virus	150	50	\$115.95	\$17,392.50
P0029	Microsoft Bookshelf	250	100	\$129.95	\$32,487.50
*(AutoNumber)					

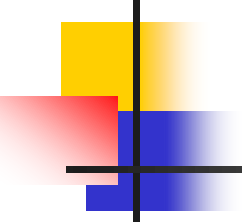


Multiple table queries--Joins

- If we run a query that uses data from multiple related tables, it essentially creates the *join* of those tables and selects records from that.
- For example, here's a query that just returns the join of the Customers and Orders tables:

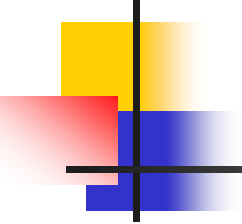


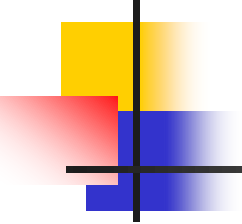
Field:	OrderID	CustomerID	OrderDate	FirstName	LastName	Address	City	State	ZipCode	PhoneNumber		
Table:	Orders	Orders	Orders	Customers	Customers	Customers	Customers	Customers	Customers	Customers		
Sort:	Ascending											
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Criteria:												
or:												

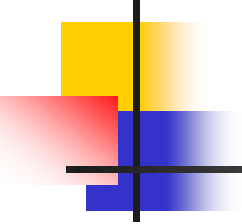
- 
-
- Since there is a many-to-one relationship between orders and customers, we can see that in this join some customer data appears multiple times (which is why a multiple-table design is preferable):

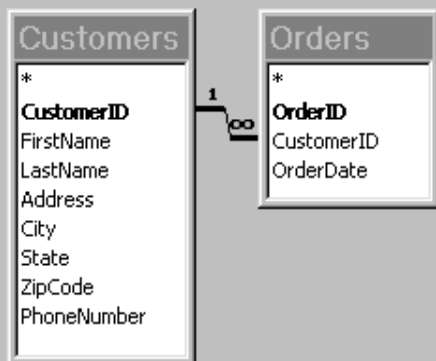


OrderID	Customer ID	Order Date	First Name	Last Name	Address	City	State	Zip Code	Phone Number
00001	C0004	4/15/99	Nicholas	Colon	9020 N.W. 75 Street	Coral Springs	FL	33065	(305) 753-9887
00002	C0003	4/18/99	Neil	Goodman	4215 South 81 Street	Margate	FL	33065	(305) 444-5555
00003	C0006	4/18/99	Jeffrey	Muddell	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
00004	C0007	4/18/99	Ashley	Geoghegan	7500 Center Lane	Coral Springs	FL	33070	(305) 753-7830
00005	C0001	4/20/99	Benjamin	Lee	1000 Call Street	Tallahassee	FL	33340	(904) 327-4124
00006	C0001	4/21/99	Benjamin	Lee	1000 Call Street	Tallahassee	FL	33340	(904) 327-4124
00007	C0002	4/21/99	Eleanor	Milgrom	7245 NW 8 Street	Margate	FL	33065	(305) 974-1234
00008	C0002	4/22/99	Eleanor	Milgrom	7245 NW 8 Street	Margate	FL	33065	(305) 974-1234
00009	C0001	4/22/99	Benjamin	Lee	1000 Call Street	Tallahassee	FL	33340	(904) 327-4124
00010	C0002	4/22/99	Eleanor	Milgrom	7245 NW 8 Street	Margate	FL	33065	(305) 974-1234
00011	C0001	4/24/99	Benjamin	Lee	1000 Call Street	Tallahassee	FL	33340	(904) 327-4124
00012	C0007	4/24/99	Ashley	Geoghegan	7500 Center Lane	Coral Springs	FL	33070	(305) 753-7830
00013	C0004	4/24/99	Nicholas	Colon	9020 N.W. 75 Street	Coral Springs	FL	33065	(305) 753-9887
00014	C0006	4/25/99	Jeffrey	Muddell	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
00015	C0009	4/25/99	Luis	Couto	455 Bargello Avenue	Coral Gables	FL	33146	(305) 666-4801
00016	C0006	4/26/99	Jeffrey	Muddell	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
00017	C0011	4/26/99	Lauren	Center	12380 S.W. 137 Avenue	Miami	FL	33186	(305) 385-4432
00018	C0011	4/26/99	Lauren	Center	12380 S.W. 137 Avenue	Miami	FL	33186	(305) 385-4432
00019	C0012	4/27/99	Robert	Slane	4508 N.W. 7 Street	Miami	FL	33131	(305) 635-3454
00020	C0012	4/28/99	Robert	Slane	4508 N.W. 7 Street	Miami	FL	33131	(305) 635-3454
00021	C0010	4/29/99	Derek	Anderson	6000 Tigertail Avenue	Coconut Grov	FL	33120	(305) 446-8900
00022	C0010	4/29/99	Derek	Anderson	6000 Tigertail Avenue	Coconut Grov	FL	33120	(305) 446-8900
00023	C0008	4/30/99	Serena	Sherard	5000 Jefferson Lane	Gainesville	FL	32601	(904) 375-6442
00024	C0006	5/1/99	Jeffrey	Muddell	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
00025	C0006	5/1/99	Jeffrey	Muddell	9522 S.W. 142 Street	Miami	FL	33176	(305) 253-3909
* oNumber)									

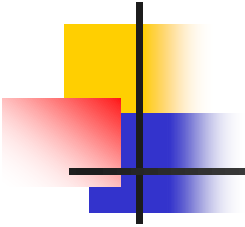
- 
-
- Now let's create a query that returns for each customer how many orders that customer has in the Orders table.
 - Conceptually, this involves grouping all the records in the join of the two tables by customer, then computing a count of how many such records there are in each group.

- 
-
- We can accomplish this by counting just the number of Order ID's there are for each customer.
 - To do grouping, we need to have the design grid include a Total: row.
 - This is done by View => Totals

- 
-
- We select Count from the drop-down menu in the Total: row under the OrderID column.
 - To see the customers listed so the ones with the largest number of orders are at the top, we also specify a descending sort order on this count field.
 - Our design is:



Field:	CustomerID	LastName	FirstName	OrderID				
Table:	Customers	Customers	Customers	Orders				
Total:	Group By	Group By	Group By	Count				
Sort:				Descending				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Criteria:								
or:								

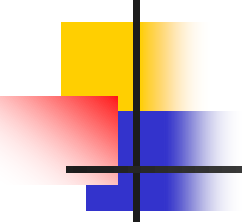


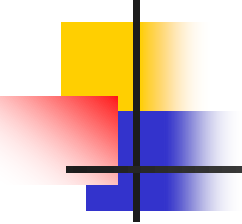
-
- The result is:

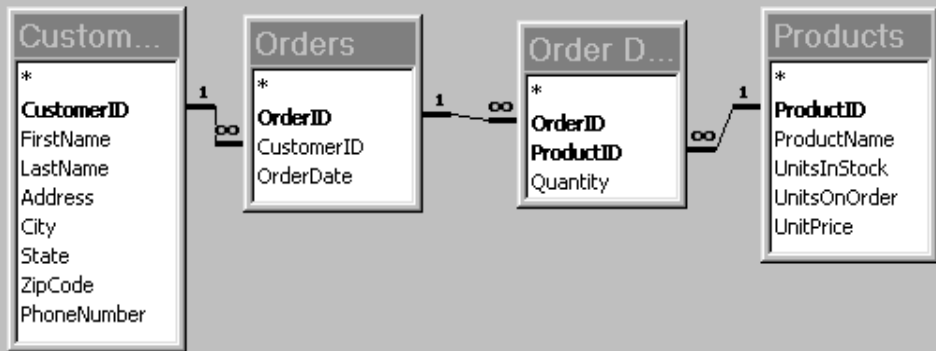


CustomerID	Last Name	First Name	CountOfOrderID
C0006	Muddell	Jeffrey	5
C0001	Lee	Benjamin	4
C0002	Milgrom	Eleanor	3
C0012	Slane	Robert	2
C0011	Center	Lauren	2
C0010	Anderson	Derek	2
C0007	Geoghegan	Ashley	2
C0004	Colon	Nicholas	2
C0009	Couto	Luis	1
C0008	Sherard	Serena	1
C0003	Goodman	Neil	1

Note the name Access gave to this new field

- 
-
- Queries like this that involve combining data from several related records are called *total* or *aggregate* queries.

- 
-
- Here is another example.
 - Suppose we want to find out how much each customer has spent, and list the results in descending order of how much each has spent buying from us.
 - Here's the query design:



Field:	CustomerID	LastName	FirstName	TotalSpent: [Quantity]*[UnitPrice]			
Table:	Customers	Customers	Customers				
Total:	Group By	Group By	Group By	Sum			
Sort:				Descending			
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Criteria:							
or:							

- 
-
- And here's the result:



CustomerID	Last Name	First Name	TotalSpent
C0002	Milgrom	Eleanor	\$23,783.75
C0006	Muddell	Jeffrey	\$8,670.27
C0007	Geoghegan	Ashley	\$8,246.50
C0001	Lee	Benjamin	\$7,293.60
C0004	Colon	Nicholas	\$5,141.26
C0003	Goodman	Neil	\$3,982.95
C0010	Anderson	Derek	\$2,414.95
C0011	Center	Lauren	\$822.70
C0012	Slane	Robert	\$819.70
C0009	Couto	Luis	\$249.90
C0008	Sherard	Serena	\$225.85



More on criteria

- By adding criteria, you can limit the groups for which you're performing a calculation, limit the records included in the calculation, or limit the results that are displayed after the calculation is performed.
- In the query design grid, where you specify criteria determines when the calculation is performed.

- To limit groupings before you perform calculations on groups of records, specify the criteria in the **Group By** fields.

This query totals extended prices for ...

Field:	ShipCountry	CompanyName	ExtendedPrice
Table:	Orders	Customers	Order Details Extended
Total:	Group By	Group By	Sum
Sort:			
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:	"Canada" Or "UK"		

Query grid

... companies in Canada and the UK only.

Ship Country	Company Name	SumOfExtendedPrice
Canada	Bottom-Dollar Markets	\$28,025.51
Canada	Laughing Bacchus	\$522.50
Canada	Mère Paillarde	\$37,123.65
UK	Around the Horn	\$14,602.15
UK	B's Beverages	\$7,383.90

Result

- To return only selected results after the calculation is performed, specify criteria in the field that contains the calculation. This rule applies whether you're performing the calculation on each record, groups of records, or all records.

Field:	ShipCountry	CompanyName	ExtendedPrice
Table:	Orders	Customers	Order Details Extended
Total:	Group By	Group By	Sum
Sort:			
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:	"Canada" Or "UK"		<10000
or:			

This query totals extended prices for companies in Canada and the UK ...

... but displays only those that are less than \$10,000.

Ship Country	Company Name	SumOfExtendedPrice
Canada	Bottom-Dollar Markets	\$28,025.51
Canada	Laughing Bacchus	\$522.50
Canada	Mère Paillarde	\$37,123.65
UK	Around the Horn	\$14,602.15
UK	B's Beverages	\$7,383.90

Ship Country	Company Name	SumOfExtendedPrice
Canada	Laughing Bacchus	\$522.50
UK	B's Beverages	\$7,383.90

- To limit the records before they are grouped and before the calculation is performed, add to the design grid the field whose records you want to limit, and then specify criteria in the field's **Criteria** cell. If you're calculating totals in the same query, set the **Total** cell for the field containing the criteria to **Where**. This rule applies whether you're calculating the total on all records or groups of records. (Microsoft Access automatically clears the **Show** check box.)

The following example uses the ExtendedPrice field twice, once to limit the records and once to calculate the total. However, you can use a different field to limit records by dragging that field to the design grid and setting its **Total** cell to **Where**.

From these records the query retrieves only those with extended prices greater than \$500.00 before it groups or totals ...

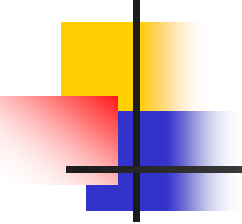
Ship Country	Company Name	ExtendedPrice
UK	Seven Seas Imports	\$945.00
Germany	Lehmanns Marktstand	\$525.00
UK	Seven Seas Imports	\$210.00
Canada	Mère Paillard	\$691.20

Field:	ShipCountry	CompanyName	ExtendedPrice	ExtendedPrice
Table:	Orders	Customers	Order Details Extended	Order Details Extended
Total:	Group By	Group By	Sum	Where
Sort:				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:	"Canada" Or "UK"			<500
or:				

... and then it sums and displays only those totals for companies in Canada and the UK.

Ship Country	Company Name	SumOfExtendedPrice
Canada	Bottom-Dollar Markets	\$11,998.00
Canada	Mère Paillard	\$21,691.36
UK	Around the Horn	\$4,907.50
UK	B's Beverages	\$1,380.00
UK	Eastern Connection	\$11,463.00
UK	Seven Seas Imports	\$9,799.60

This total for Seven Seas Imports does not include the order for \$210.00.

- 
-
- Following are some of the operators that can appear in expressions within design grid cells.
 - First, the use of square brackets around a field name, like [UnitPrice], in an expression, means to use the field's value in that expression.



Numerical comparison operators

- $<$
- $<=$
- $>$
- $>=$
- $=$
- $<>$
- Between
- Less than
- Less than or equal
- Greater than
- Greater than or equal
- Equal
- Not equal
- Within the range (inclusive)



Arithmetic operators

- \wedge (exponentiation)
- $*$ (multiplication)
- $/$ (division)
- \backslash (division that returns an integer result)
- Mod (divide two numbers and return only the remainder)
- $+$ (addition)
- $-$ (subtraction)



'Like' and 'Is' comparison operators

- The Like operator is a pattern-matching operator used to compare two strings.
- With Like, the single-character wild-card '?' and the arbitrary-length wild-card '*' can be used, as in 'Like A*'.
The 'Like' operator is used to compare a string against a pattern. The '?' wildcard matches any single character, and the '*' wildcard matches any sequence of characters.
- The Is operator is most often used in the context Is Null, to test whether a field is blank. (Is Not Null is satisfied in just the opposite case.)
The 'Is' operator is used to test for null values. 'Is Null' returns true if the field is empty, while 'Is Not Null' returns true if the field contains a value.



Logical operators

- And
 - Used to perform a logical conjunction on two expressions.
- Or
 - Used to perform a logical disjunction on two expressions.
- Not
 - Used to perform a logical negation on an expression.



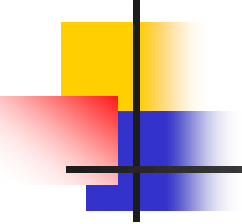
The 'In' operator

- Determines whether the value of an expression is equal to any of several values in a specified list.



Parameter queries

- A parameter query prompts for criteria each time it's run.
- A parameter query displays one or more predefined dialog boxes that prompt you for the parameter value (criteria).
- You can also create a custom dialog box that prompts for the query's parameters.

- 
-
- For example, suppose we often want to see certain data for all customers living in a particular city, where that city may be Miami, or Coral Gables, or any one of many other cities.
 - Rather than create a separate query with the city built in, it may make more sense to create a single parameter query that prompts for the city before it runs.



Creating a parameter query

- In the Criteria cell for each field you want to use as a parameter, type a prompt enclosed in square brackets. Microsoft Access will display this prompt when the query is run. The text of the prompt must be different from the field name, although it can include the field name.



Microsoft Access 97



Help Topics

Back

Options

Field:

LastName

Table:

Employees

Sort

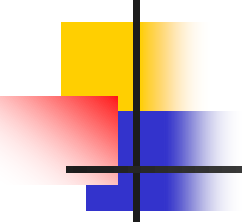
Show:



Criteria:

[Type the last name:]

The prompt "Type the last name:" will be displayed when you run the query.

- 
-
- For example, a simple modification of our earlier query to list all customer data for Miami customers makes it a parameter query that prompts for the city:

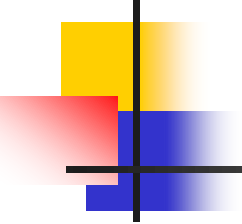


Customers

*

- CustomerID
- FirstName
- LastName
- Address
- City
- State
- ZipCode
- PhoneNumber

Field:	CustomerID	LastName	FirstName	Address	City	State	ZipCode	PhoneNumber	
Table:	Customers	Customers	Customers	Customers	Customers	Customers	Customers	Customers	
Sort:									
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
Criteria:					[What city?]				
or:									

- 
-
- For a field that displays dates, you can display the prompts "Type the beginning date:" and "Type the ending date:" to specify a range of values. In the field's Criteria cell, type Between [Type the beginning date:] And [Type the ending date:].



Crosstab queries

- A crosstab query displays summarized values (sums, counts, and averages) from one field in a table and groups them by one set of facts listed down the left side of the datasheet and another set of facts listed across the top of the datasheet.

Example of a crosstab query

Crosstab queries calculate a sum, average, count, or other type of total for data that is grouped by two types of information — one down the left side of the datasheet and another across the top.

Last Name	Category Name	SumOfSubtotal
Buchanan	Beverages	\$46,302.09
Buchanan	Condiments	\$16,789.95
Buchanan	Confections	\$36,182.13
Callahan	Beverages	\$111,047.76
Callahan	Condiments	\$49,566.21
Callahan	Confections	\$80,005.35

This select query only groups the totals vertically by employee and category. This results in more records, making comparisons between different employees' totals more difficult.

Last Name	Beverages	Condiments	Confections
Buchanan	\$46,302.09	\$16,789.95	\$36,182.13
Callahan	\$111,047.76	\$49,566.21	\$80,005.35
Davolio	\$135,575.82	\$57,312.91	\$103,961.31
Dodsworth	\$55,931.04	\$37,270.00	\$30,342.63
Fuller	\$135,209.93	\$51,675.89	\$82,459.47
King	\$105,355.83	\$38,878.32	\$72,023.27

A crosstab query displays the same information, but groups it both horizontally and vertically so the datasheet is more compact.

Example of a crosstab query

You create a crosstab query with a wizard or from scratch in the query design grid. In the design grid, you specify which field's values become column headings, which field's values become row headings, and which field's values to sum, average, count, or otherwise calculate.



In query Design view, click the Query Type button, click Crosstab, and then...

...click the settings you want to use in these rows.

Field:	LastName	CategoryName	Subtotal
Table:	Employees	Categories	Order Subtotals
Total:	Group By	Group By	Sum
Crosstab:	Row Heading	Column Heading	Value
			Row Heading
			Column Heading
			Value
			(not shown)

This setting uses the field's values as row headings.

This setting displays the field's values as column headings.

These settings display the total orders.

Last Name	Beverages	Condiments	Confections
Buchanan	\$46,302.09	\$16,789.95	\$36,182.13
Callahan	\$111,047.76	\$49,566.21	\$80,005.35
Davolio	\$135,575.82	\$57,312.91	\$103,961.31



To create a crosstab query using a wizard

- In the Database window, click the Queries tab, and then click New.
- In the New Query dialog box, click Crosstab Query Wizard.
- Click OK.
- Follow the directions in the wizard dialog boxes. In the last dialog box, you can choose to run the query or see the query's structure in Design view.



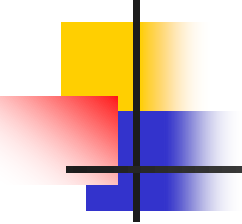
Action queries

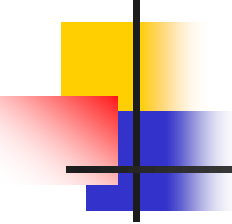
- An action query is a query that makes changes to many records in just one operation.
- There are four types of action queries:
 - delete,
 - update,
 - append, and
 - make-table.



Delete query

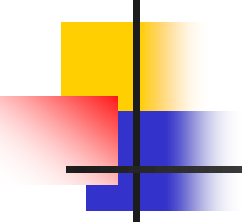
- A delete query deletes a group of records from one or more tables.
- For example, you could use a delete query to remove products that are discontinued or for which there are no orders.
- With delete queries, you always delete entire records/rows, not just selected fields within records.

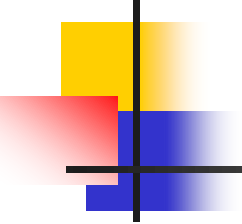
- 
-
- You can use a single delete query to delete records from a single table, from multiple tables in a one-to-one relationship, or from multiple tables in a one-to-many relationship, if cascading deletes are enabled (for example, all customers from Miami and all their orders).

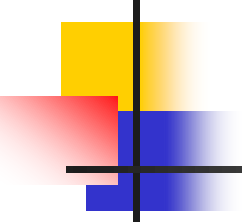


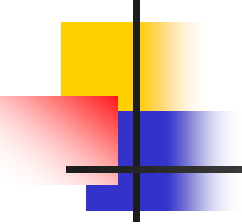
Simplest case: Delete records from one table or tables in a one-to-one relationship

- 1 Create a new query that contains the tables from which you want to delete records.
- 2 In query Design view, click the arrow next to Query Type on the toolbar, and then click Delete Query.

- 
-
- 3 For the tables you want to delete records from, drag the asterisk (*) from the field list to the query design grid.
 - **From** appears in the Delete cell under these fields.

- 
-
- 4 To specify criteria for deleting records, drag to the design grid the fields on which you want to set criteria.
 - **Where** appears in the Delete cell under these fields.

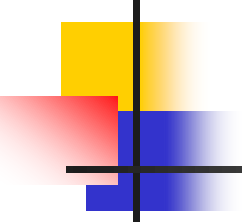
- 
-
- 5 In the Criteria cell for the fields that you have dragged to the grid, type the criteria.
 - 6 To preview the records that will be deleted, click View on the toolbar. To return to query Design view, click View on the toolbar again. Make any changes you want in Design view.

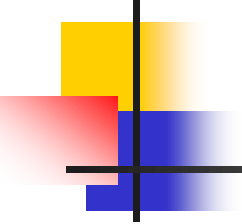
- 
-
- 7 Click Run on the toolbar to delete the records.
 - Note: To stop a query after you start it, press CTRL+BREAK.

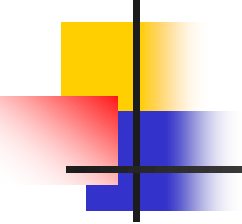


Important considerations when using a query that deletes records

- **Once you delete records using a delete query, you can't undo the operation.** Therefore, you should preview the data that the query selected for deletion *before* you run the query. You can do this by clicking View on the toolbar, and viewing the query in Datasheet view.

- 
-
- You should maintain backup copies of your data at all times.
 - If you delete the wrong records, you can retrieve them from your backup copies.

- 
-
- In some cases, running a delete query might delete records in related tables, even if they're not included in the query. This can happen when your query contains only the table that's on the "one" side of a one-to-many relationship, and you've enabled cascading deletes for that relationship.

- 
-
- In that case when you delete records from the "one" table, you'll also delete records from the "many" table.



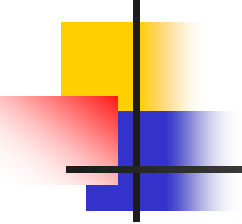
Update query

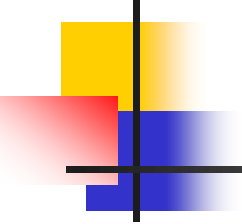
- An update query makes global changes to a group of records in one or more tables.
- For example, you can raise prices by 10 percent for all products, or you can raise salaries by 5 percent for the people within a certain job category. With an update query, you can change data in existing tables.

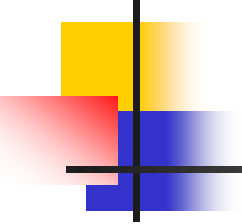


Change records as a group using an update query

- 1 Create a query, selecting the tables or queries that include the records you want to update and the fields you want to use for setting criteria.
- 2 In query Design view, click the arrow next to Query Type on the toolbar, and then click Update Query.

- 
-
- 3 Drag from the field list to the query design grid the fields you want to update or you want to specify criteria for.
 - 4 In the Criteria cell, specify the criteria if necessary.

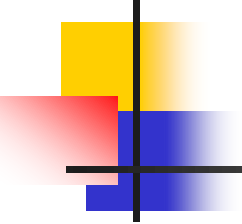
- 
-
- 5 In the Update To cell for the fields you want to update, type the expression or value you want to use to change the fields.
 - 6 To see a list of the records that will be updated, click View on the toolbar. This list won't show the new values. To return to query Design view, click View on the toolbar again. Make any changes you want in Design view.

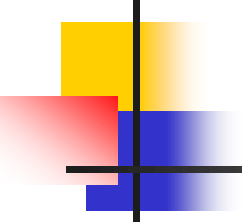
- 
-
- 7 Click Run on the toolbar to create the new table.
 - Note: As usual, to stop a query after you start it, press CTRL+BREAK.



Append query

- An append query adds a group of records from one or more tables to the end of one or more tables.
- For example, suppose that you acquire some new customers and a database containing a table of information on those customers. To avoid typing all this information in, you'd like to append it to your Customers table.

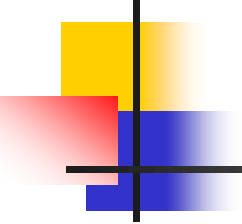
- 
-
- Append queries are also helpful for:
 - Appending fields based on criteria. For example, you might want to append only the names and addresses of customers with outstanding orders.

- 
-
- Sometimes you want to append records when some of the fields in one table don't exist in the other table. For example, a Customers table has 11 fields. Suppose that you want to append records from another table that has fields that match 9 of the 11 fields in the Customers table. An append query will append the data in the matching fields and ignore the others.



Make-table query

- A make-table query creates a new table from all or part of the data in one or more tables.
- Make-table queries are helpful for:
- Creating a table to export to other Microsoft Access databases.
- For example, you might want to create a table that contains several fields from your Employees table, and then export that table to a database used by your personnel department.

- 
-
- Make-table queries are also useful for making a backup copy of a table and for
 - Creating a history table that contains old records.
 - For example, you could create a table that stores all your old orders before deleting them from your current Orders table.