# Access 2010: Action Queries

## Table of Contents

**About Action Queries** .......................................................................................................................... 2  
**Update Query** ..................................................................................................................................... 2  
**Append Query** ..................................................................................................................................... 2  
**Make-Table Query** ............................................................................................................................. 2  
**Delete Query** ...................................................................................................................................... 2  
  - Important considerations when using a delete query ........................................................................ 2  

## Add Records to a Table by Using an Append Query ......................................................................... 3  
**Benefits of Using an Append Query** ................................................................................................. 3  
**Basic Steps of an Append Query** ...................................................................................................... 3  
**Create and Run an Append Query** .................................................................................................... 4  
  - Step 1: Create a query to select the records to copy ..................................................................... 4  
  - Step 2: Convert the select query to an append query .................................................................... 4  
  - Step 3: Choose the destination fields ............................................................................................ 5  
  - Step 4: Preview and run the append query ..................................................................................... 5  

## Update Data by Using a Query ........................................................................................................... 6  
**Similarities and Differences Between Find and Replace and an Update Query** ........................... 6  
**Restrictions on Fields That Can be Updated** ................................................................................... 6  
**Create and Run an Update Query** .................................................................................................... 6  
  - Step 1: Create a select query to identify the records to update ....................................................... 7  
  - Step 2: Update the records ............................................................................................................. 7  
**Update Data From One Table to Another** ........................................................................................ 8  
  - Create and run the update query ..................................................................................................... 9  
**Restrictions on Data Type Conversions** ........................................................................................... 9  

## Create a Make Table Query .................................................................................................................. 12  
**Create a Make Table Query** ............................................................................................................ 13  
  - Create the select query .................................................................................................................. 13  
  - Convert the select query ................................................................................................................ 13  

## Delete Data from an Access Database by Using a Query ................................................................ 14  
**Choose a Query Type Based on the Data You Want to Delete** ....................................................... 14  
  - What to verify before using a query to delete data ..................................................................... 14  
**Create and Use a Delete Query** ....................................................................................................... 15  
**Deleting Data from Related Tables** .................................................................................................. 15  
**Why Am I Seeing This Error Message, and How Do I Fix It?** .......................................................... 17  

## Additional Information ......................................................................................................................... 17  
**Back Up the Database** ..................................................................................................................... 17  
**Stop Disabled Mode from Blocking a Query** ................................................................................... 17  

## Table of Example Criteria .................................................................................................................... 18
About Action Queries
An action query is a query that makes changes to or moves many records in just one operation. There are four types of action queries: append, update, make-table, and delete.

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 dairy 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.

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.

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.
- **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.
- **Improving performance of forms and reports based on multiple-table queries**: For example, suppose you want to print multiple reports that are based on a five-table query that includes totals. You might be able to speed things up by first creating a make-table query that retrieves the records you need and stores them in one table. Then you can base the reports on this table as the record source for a form or report so you don't have to rerun the query each time you open the form or report. However, the data in the table is frozen at the time you run the make-table query.

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, not just selected fields within records.

You can use a 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, you could use a delete query to delete all customers from Ireland and all their orders. However, if you need to include the "many" table along with the "one" table in order to add criteria, you must run the query twice, since a query cannot delete records from the primary table and the related tables at the same time.

You can distinguish a one-to-many relationship in query Design view by looking at the joins between tables. If one end of a join is marked with the infinity symbol [∞], it is a one-to-many relationship. If both ends are marked with a 1, it's a one-to-one relationship.

Important considerations when using a delete query
- 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. When you delete records from the "one" table, you'll also delete records from the "many" table.

• When a delete query contains more than one table, such as a query that deletes duplicate records from one of the tables, the query’s UniqueRecords property must be set to Yes.

Add records to a table by using an append query
You use an append query when you need to add new records to an existing table by using data from other sources.

If you need to change data in an existing set of records, such as updating the value of a field, you can use an update query. If you need to make a new table from a selection of data, or to merge two tables into one new table, you can use a make-table query.

Benefits of using an append query
By using a query to copy data, you can:

• Append multiple records in one pass: If you copy data manually, you usually have to perform multiple copy/paste operations. By using a query, you select all the data at once, and then copy it.

• Review your selection before you copy it: You can view your selection in Datasheet view and can make adjustments to your selection as needed before you copy the data. This can be particularly handy if your query includes criteria or expressions, and you need several tries to get it just right. You cannot undo an append query. If you make a mistake, you must either restore your database from a backup or correct your error, either manually or by using a delete query.

• Use criteria to refine your selection: For example, you might want to only append records of customers who live in a specific city.

• Append records when some of the fields in the data sources don't exist in the destination table: For example, suppose that your existing customer table has eleven fields, and the new table that you want to copy from only has nine of those eleven fields. You can use an append query to copy the data from the nine fields that match and leave the other two fields blank.

Basic steps of an append query
The process of creating an append query follows these basic steps:

• Create a select query: You start by selecting the data that you want to copy. You can adjust your select query as needed, and run it as many times as you want to make sure you are selecting the data that you want to copy.

• Convert the select query to an append query: After your selection is ready, you change the query type to Append.

• Choose the destination fields for each column in the append query: In some cases, Access automatically chooses the destination fields for you. You can adjust the destination fields, or choose them if Access did not.

• Preview and run the query to append the records: Before you append the records, you can switch to Datasheet view for a preview of the appended records.
Create and run an append query

Important: *You cannot undo an append query*. Consider backing up your database or the destination table.

**Step 1: Create a query to select the records to copy**

1. Open the database that contains the records that you want to copy.
2. On the **Create** tab, in the **Queries** group, click **Query Design**.
   
   a) The query designer opens, and the **Show Table** dialog box appears.
3. Double-click the tables or queries that contain the records that you want to copy, and then click **Close**.
   
   a) The tables or queries appear as one or more windows in the query designer. Each window lists the fields in a table or query. The figure on the right shows a typical table in the query designer.
4. Double-click each field that you want to append.
   
   a) The selected fields appear in the **Field** row in the query design grid.

- The data types of the fields in the source table must be compatible with the data types of the fields in the destination table. Text fields are compatible with most other types of fields. Number fields are only compatible with other number fields. For example, you can append numbers to a text field, but you cannot append text into a number field.
- You can also use an expression as a field (such as `=Date()` to automatically return today's date) and you can use expressions with table or query fields in the design grid to customize the data that you select. For example, if your destination table has a field that stores a four-digit year and the source table has a regular date/time field, you might use the **DatePart** function with the source field to select only the year.

5. To quickly add all the fields in a table, double-click the asterisk (*) at the top of the list of table fields. The figure on the right shows the design grid with all fields added.
   
   a) Optionally, you can enter one or more criteria in the **Criteria** row of the design grid.
6. On the **Design** tab, in the **Results** group, click **Run**.
7. Verify that the query returned the records that you want to copy. If you need to add or remove fields from the query, switch back to Design view and add fields as described in the preceding step, or select the fields that you don't want and press DELETE to remove them from the query.

**Step 2: Convert the select query to an append query**

1. On the **Home** tab, in the **View** group, click **View**, and then click **Design View**.
2. On the **Design** tab, in the **Query Type** group, click **Append**.
   
   a) The **Append** dialog box appears.
3) Next, you specify whether to append records to a table in the current database, or to a table in a different database. Do one of the following:
   a) In the **Append** dialog box, click **Current Database**, select the destination table from the **Table Name** combo box, and then click **OK**.
   -or-
   b) In the **Append** dialog box, click **Another Database**.
      - In the **File Name** box, enter the location and name of the destination database.
      - In the **Table Name** combo box, enter the name of the destination table, and then click **OK**.

**Step 3: Choose the destination fields**
The way that you choose destination fields depends on how you created your select query in Step 1.

<table>
<thead>
<tr>
<th>If you...</th>
<th>Access...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Added all the fields from your source table or query</td>
<td>Adds all the fields in the destination table to the <strong>Append to</strong> row in the design grid</td>
</tr>
<tr>
<td>Added individual fields to the query or used expressions, and the field names in the source and destination tables match</td>
<td>Automatically adds the matching destination fields to the <strong>Append to</strong> row in the query</td>
</tr>
<tr>
<td>Added individual fields or used expressions, and any of the names in the source and destination tables don't match</td>
<td>Adds the matching fields, and leaves unmatched fields blank</td>
</tr>
</tbody>
</table>

If Access leaves fields blank, you can click a cell in the **Append to** row and select a destination field.

The figure illustrates how you click a cell in the **Append to** row and select a destination field.

**Note:** If you leave the destination field blank, the query will not append data to that field.

**Step 4: Preview and run the append query**
1) To preview your changes, switch to Datasheet view.
   **Tip:** To quickly switch views, right-click the tab at the top of the query, and then click the view that you want.
2) Return to Design view, and then click **Run ✓** to append the records.
   **Important:** You cannot undo the results of an append query.
Update data by using a query
You use update queries to add, change, or delete the data in one or more existing records. You can think of update queries as a powerful form of the **Find and Replace** dialog box.

You cannot use an update query to add new records to a database, or to delete entire records from a database. To add new records to a database you use an append query, and to delete whole records from a database you use a delete query.

**Similarities and differences between Find and Replace and an update query**
Like the **Find and Replace** dialog box, an update query lets you:
- Specify a value that you want to replace.
- Specify the value to use as a replacement.

Unlike the **Find and Replace** dialog box, an update query lets you:
- Use criteria that do not depend on the value that you want to replace.
- Update lots of records in one pass.
- Change records in more than one table at the same time.

**Restrictions on fields that can be updated**
An update query cannot be used to update data in the following types of fields:
- **Calculated fields:** The values in calculated fields do not permanently reside in tables. They only exist in your computer's temporary memory after Access calculates them. Because calculated fields do not have a permanent storage location, you cannot update them.
- **Fields from a totals query or a crosstab query:** The values in these types of query are calculated, and therefore cannot be updated by an update query.
- **AutoNumber fields:** By design, the values in AutoNumber fields change only when you add a record to a table.
- **Fields in unique-values queries and unique-records queries:** The values in such queries are summarized. Some of the values represent a single record, and others represent more than one record. The update operation is not possible because it is not possible to determine what records were excluded as duplicates, and therefore it is not possible to update all the necessary records. This restriction applies whether you use an update query or try to update data manually by entering values in a form or a datasheet.
- **Fields that are primary keys:** In some cases, such as if the primary key field is used in a table relationship, you cannot update the field by using a query unless you first set the relationship to automatically cascade updates.

**Note:** When you cascade updates, Access automatically updates foreign key values when you change a primary key value in a parent table.

**Create and run an update query**
As a best practice, there are two main steps that you must follow to create and run an update query: create a select query that identifies the records that you want to update, and then convert that query to an update query that you can run to update the records.

**Tip:** You might want to back up your database before you run an update query. You cannot undo the results of an update query, and making a backup helps make sure that you can reverse your changes if you change your mind.
**Step 1: Create a select query to identify the records to update**

1) Open the database that contains the records you want to update.

2) On the **Create** tab, in the **Queries** group, click **Query Design**.
   a) The query designer opens, and the **Show Table** dialog box opens.

3) Click the **Tables** tab.

4) Select the table or tables that contain the records that you want to update, click **Add**, and then click **Close**.
   a) The table or tables appear as one or more windows in the query designer, and the windows list all the fields in each table.

5) Double-click the fields that you want to update in the table windows.
   a) The selected fields appear in the **Field** row in the query design grid.
   b) You can add one table field per column in the query design grid.
   c) To add all the fields in a table quickly, double-click the asterisk (*) at the top of the list of table fields in the table window.

6) To limit the query results based on field values, in the query design grid, in the **Criteria** row, enter the criteria that you want to use to limit the results.

7) On the **Design** tab, in the **Results** group, click **Run**.

8) Verify that the query returns the records that you want to update.
   a) To remove any fields that you do not want included in the query design, select the fields and then press **DELETE**.
   b) To add any fields that you want to include in the query design, drag the additional fields to the query design grid.

**Step 2: Update the records**

This procedure shows you how to change a select query to an update query. When you do this, Access adds the **Update to** row in the query design grid.

1) In Design view, on the **Design** tab, in the **Query Type** group, click **Update**.

2) Locate the field that contains the data that you want to change, and then type your expression (your change criteria) in the **Update to** row for that field.

   You can use any valid expression in the **Update to** row.

   The illustration shows an update query that returns all the assets purchased after January 5, 2005 and changes the location to "Warehouse 3" for all the records that meet that criterion. See the next page for example expressions.

3) On the **Design** tab, in the **Results** group, click **Run**.

   a) An alert message appears.

4) To run the query and update the data, click **Yes**.

**Note:** When you run the query, you might notice that some fields are missing from your result set. If your query contains fields that you don't update, Access does not display those fields in the results, by default. For example, you might include ID fields from two tables to help ensure that your query identifies and updates the correct records. If you don't update those ID fields, Access does not display them in the results.
Table of example expressions
This table shows some example expressions and explains how they change data.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Salesperson&quot;</td>
<td>In a Text field, changes a text value to Salesperson.</td>
</tr>
<tr>
<td>#8/10/07#</td>
<td>In a Date/Time field, changes a date value to 10-Aug-07.</td>
</tr>
<tr>
<td>Yes</td>
<td>In a Yes/No field, changes a No value to Yes.</td>
</tr>
<tr>
<td>&quot;PN&quot; &amp; [PartNumber]</td>
<td>Adds &quot;PN&quot; to the beginning of each specified part number.</td>
</tr>
<tr>
<td>[UnitPrice] * [Quantity]</td>
<td>Multiplies the values in fields named UnitPrice and Quantity.</td>
</tr>
<tr>
<td>[Freight] * 1.5</td>
<td>Increases the values in a field named Freight by 50 percent.</td>
</tr>
<tr>
<td>DSum(&quot;[Quantity] * [UnitPrice]&quot;, &quot;Order Details&quot;, &quot;[ProductID]=&quot; &amp; [ProductID])</td>
<td>Where the ProductID values in the current table match the ProductID values in table named Order Details, this expression updates sales totals by multiplying the values in a field named Quantity by the values in a field named UnitPrice. The expression uses the DSum function because it can operate against more than one table and table field.</td>
</tr>
<tr>
<td>Right([ShipPostalCode], 5)</td>
<td>Truncates (removes) the leftmost characters in a text or numeric string and leaves the 5 rightmost characters.</td>
</tr>
<tr>
<td>IIf(IsNull([UnitPrice]), 0, [UnitPrice])</td>
<td>Changes a null (unknown or undefined) value to a zero (0) value in a field named UnitPrice.</td>
</tr>
</tbody>
</table>

Update data from one table to another
When you need to update data from one table to another, consider the following rule: the data types for the source and destination fields must either match or be compatible.

Furthermore, when you update data from one table to another and use compatible data types instead of matching data types, Access converts the data types of those fields in the destination table. As a result, some of the data in the destination fields may be truncated (deleted). The section Restrictions on data type conversions lists the ways in which you can and cannot convert data types.

The process of updating data from one table to another follows these broad steps:
1) Create an update query and add both the source and destination tables to the query.
2) Join those tables on the fields that contain the related information.
3) Add the names of your destination fields to the Field row of the query design grid.
4) Add the names of your source fields to the Update To row of the query design grid by using the following syntax: [source_table].[source_field].
Create and run the update query

1) In Design view, on the Create tab, in the Queries group, click Query Design.
2) In the Show Table dialog box, click the Tables tab.
3) Double-click your source and destination tables to add them to the query, and then click Close.
   a) Each table appears in a window in the query designer.
4) In most cases, Access automatically joins related fields in a query. To manually join fields that contain related information, drag the related field from one table to the equivalent field in the other table.
5) On the Design tab, in the Query Type group, click Update.
6) In the destination table, double-click the fields that you want to update.
   a) Each field appears in the Field row in the query design grid.
7) In the Update To row of the query, in each of the columns that contains a destination field, add the name of the source table and the field in the source table that corresponds to the field in the destination table, and make sure that you use this syntax: [Table].[Field], where you enclose table and field names with square brackets, and you separate table and field names with a period.

As you continue, remember that you must spell the table and field names in the Update To row correctly, and you must match any punctuation in the original table and field names. However, you do not have to match capitalization.

8) On the Design tab, in the Results group, click Run.
9) When asked to confirm the update, click Yes.

Restrictions on data type conversions
The following table lists the data types that Access provides, explains any restrictions on how to convert data types, and briefly describes any data loss that might occur during conversion.

<table>
<thead>
<tr>
<th>Convert to this type</th>
<th>From this type</th>
<th>Changes or restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>Memo</td>
<td>Access deletes all but the first 255 characters.</td>
</tr>
<tr>
<td></td>
<td>Number</td>
<td>No restrictions.</td>
</tr>
<tr>
<td></td>
<td>Date/Time</td>
<td>No restrictions.</td>
</tr>
<tr>
<td></td>
<td>Currency</td>
<td>No restrictions.</td>
</tr>
<tr>
<td></td>
<td>AutoNumber</td>
<td>No restrictions.</td>
</tr>
<tr>
<td></td>
<td>Yes/No</td>
<td>The value -1 (Yes in a Yes/No field) converts to Yes. The value 0 (No in a Yes/No fields) converts to No.</td>
</tr>
<tr>
<td></td>
<td>Hyperlink</td>
<td>Access truncates links longer than 255 characters.</td>
</tr>
<tr>
<td>Convert to this type</td>
<td>From this type</td>
<td>Changes or restrictions</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Memo</td>
<td>Text</td>
<td>No restrictions.</td>
</tr>
<tr>
<td>Number</td>
<td>No restrictions.</td>
<td></td>
</tr>
<tr>
<td>Date/Time</td>
<td>No restrictions.</td>
<td></td>
</tr>
<tr>
<td>Currency</td>
<td>No restrictions.</td>
<td></td>
</tr>
<tr>
<td>AutoNumber</td>
<td>No restrictions.</td>
<td></td>
</tr>
<tr>
<td>Yes/No</td>
<td>The value -1 (Yes in a Yes/No field) converts to Yes. The value 0 (No in a Yes/No fields) converts to No.</td>
<td></td>
</tr>
<tr>
<td>Hyperlink</td>
<td>No restrictions.</td>
<td></td>
</tr>
<tr>
<td><strong>Number</strong></td>
<td>Text</td>
<td>Text must consist of numbers, valid currency, and decimal separators. The number of characters in the Text field must fall within the size set for the Number field.</td>
</tr>
<tr>
<td></td>
<td>Memo</td>
<td>The Memo field must contain only numbers and valid currency and decimal separators. The number of characters in the Memo field must fall within the size set for the Number field.</td>
</tr>
<tr>
<td></td>
<td>Number, but with a different field size or precision</td>
<td>Values must not be larger or smaller than what the new field size can store. Changing precision might cause Access to round some values.</td>
</tr>
<tr>
<td></td>
<td>Date/Time</td>
<td>The dates that you can convert depend on the size of the number field. Remember that Access stores all dates as serial dates, and it stores the date values as double-precision floating integers. Access uses December 30, 1899 as date 0. Dates outside the range April 18, 1899 and September 11, 1900 exceed the size of a Byte field. Dates outside the range April 13, 1810 and September 16, 1989 exceed the size of an Integer field. To accommodate all possible dates, set the <strong>Field Size</strong> property of your Number field to <strong>Long Integer</strong> or greater.</td>
</tr>
<tr>
<td></td>
<td>Currency</td>
<td>Values must not exceed (or fall below) the size limit set for the field. For example, you can convert a Currency field to an Integer field only when those values are greater than 255 and do not exceed 32,767.</td>
</tr>
<tr>
<td></td>
<td>AutoNumber</td>
<td>Values must fall within the size limit set for the field.</td>
</tr>
<tr>
<td></td>
<td>Yes/No</td>
<td>&quot;Yes&quot; values convert to -1. &quot;No&quot; values convert to 0.</td>
</tr>
<tr>
<td>Convert to this type</td>
<td>From this type</td>
<td>Changes or restrictions</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Date/Time</strong></td>
<td>Text</td>
<td>Original text must be a recognizable date or date/time combination. For example, 18-Jan-2007.</td>
</tr>
<tr>
<td></td>
<td>Memo</td>
<td>Original text must be a recognizable date or date/time combination. For example, 18-Jan-2007.</td>
</tr>
<tr>
<td></td>
<td>Number</td>
<td>Value must fall between -657,434 and 2,958,465.99998843.</td>
</tr>
<tr>
<td></td>
<td>Currency</td>
<td>Value must fall between -$657,434 and $2,958,465.9999.</td>
</tr>
<tr>
<td></td>
<td>AutoNumber</td>
<td>Value must exceed -657,434 and be less than 2,958,466.</td>
</tr>
<tr>
<td></td>
<td>Yes/No</td>
<td>The value -1 (Yes) converts to December 29, 1899. The value 0 (No) converts to midnight (12:00 AM).</td>
</tr>
<tr>
<td><strong>Currency</strong></td>
<td>Text</td>
<td>Text must consist of numbers and valid separators.</td>
</tr>
<tr>
<td></td>
<td>Memo</td>
<td>Text must consist of numbers and valid separators.</td>
</tr>
<tr>
<td></td>
<td>Number</td>
<td>No restrictions.</td>
</tr>
<tr>
<td></td>
<td>Date/Time</td>
<td>No restrictions, but Access might round the value</td>
</tr>
<tr>
<td></td>
<td>AutoNumber</td>
<td>No restrictions.</td>
</tr>
<tr>
<td></td>
<td>Yes/No</td>
<td>The value -1 (Yes) converts to $1, and the value 0 (No) converts to 0$.</td>
</tr>
<tr>
<td><strong>AutoNumber</strong></td>
<td>Text</td>
<td>Not allowed if the AutoNumber field serves as a primary key.</td>
</tr>
<tr>
<td></td>
<td>Memo</td>
<td>Not allowed if the AutoNumber field serves as a primary key.</td>
</tr>
<tr>
<td></td>
<td>Number</td>
<td>Not allowed if the AutoNumber field serves as a primary key.</td>
</tr>
<tr>
<td></td>
<td>Date/Time</td>
<td>Not allowed if the AutoNumber field serves as a primary key.</td>
</tr>
<tr>
<td></td>
<td>Currency</td>
<td>Not allowed if the AutoNumber field serves as a primary key.</td>
</tr>
<tr>
<td></td>
<td>Yes/No</td>
<td>Not allowed if the AutoNumber field serves as a primary key.</td>
</tr>
<tr>
<td><strong>Yes/No</strong></td>
<td>Text</td>
<td>Original text must consist only of Yes, No, True, False, On, or Off.</td>
</tr>
<tr>
<td></td>
<td>Memo</td>
<td>Original text must consist only of Yes, No, True, False, On, or Off.</td>
</tr>
<tr>
<td></td>
<td>Number</td>
<td>Zero or Null converts to No, all other values convert to Yes.</td>
</tr>
<tr>
<td></td>
<td>Date/Time</td>
<td>Null or 12:00:00 AM converts to No, all other values convert to Yes.</td>
</tr>
<tr>
<td></td>
<td>Currency</td>
<td>Zero and Null convert to No, all other values convert to Yes.</td>
</tr>
<tr>
<td></td>
<td>AutoNumber</td>
<td>All values convert to Yes.</td>
</tr>
<tr>
<td>Convert to this type</td>
<td>From this type</td>
<td>Changes or restrictions</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Hyperlink</td>
<td>Text</td>
<td>If the original text contains a valid Web address, such as adatum.com, <a href="http://www.adatum.com">www.adatum.com</a>, or <a href="http://www.adatum.com">http://www.adatum.com</a>, Access converts the text to a hyperlink. Access tries to convert other values, meaning that you see underlined text, and the cursor changes when you point at the link, but the links don't work. The text can contain any valid Web protocol, including http://, gopher://, telnet://, ftp://, and wais://.</td>
</tr>
<tr>
<td>Memo</td>
<td>See the previous entry. The same restrictions apply.</td>
<td></td>
</tr>
<tr>
<td>Number</td>
<td>Not allowed when a Number field is part of a relationship. If the original value is in the form of a valid Internet Protocol (IP) address (four number triplets separated by a period: nnn.nnn.nnn.nnn) and the numbers happen to coincide with a Web address, the conversion results in a valid link. Otherwise, Access appends http:// to the beginning of each value, and the resulting links are not valid.</td>
<td></td>
</tr>
<tr>
<td>Date/Time</td>
<td>Access appends http:// to the beginning of each address, but the resulting links will almost never work.</td>
<td></td>
</tr>
<tr>
<td>Currency</td>
<td>Access appends http:// to the beginning of each value, but like dates, the resulting links will almost never work.</td>
<td></td>
</tr>
<tr>
<td>AutoNumber</td>
<td>Not allowed when the AutoNumber field is part of a relationship. Access appends http:// to the beginning of each value, but the resulting links will almost never work.</td>
<td></td>
</tr>
<tr>
<td>Yes/No</td>
<td>Access converts all Yes values to -1 and all No values to 0, and appends http:// to the beginning of each value. The resulting links do not work.</td>
<td></td>
</tr>
</tbody>
</table>

**Create a make table query**

You use a make table query when you need to copy the data in a table or archive data. If you need to change or update part of the data in an existing set of records, such as one or more fields, you can use an update query. If you need to add records (rows) to an existing table, you use an append query.

A make table query retrieves data from one or more tables, and then loads the result set into a new table. That new table can reside in the database that you have open, or you can create it in another database. Typically, you create make table queries when you need to copy or archive data. As you proceed, remember that the data in your new table is strictly a snapshot; it has no relationship or connection to its source table or tables.

The process of creating a make table query follows these broad steps:

- Enable the database if it is not signed or if it does not reside in a trusted location. You cannot run action queries (append, update, and make table queries) otherwise.
- In query Design view, create a select query and then modify that query until it returns the records you want. You can select data from more than one table and, in a real sense, you can de-normalize your data. You can also use criteria in the query to further customize or narrow your result set.
• Convert the select query to a make table query, choose a location for the new table, and then run the query to create the table.

Create a make table query
You create a make table query by first creating a select query, and then converting it to a make table query. Your select query can use calculated fields and expressions to help return the data that you need. The following steps explain how to create and convert the query. If you already have a select query that fits your needs, you can skip ahead to the steps for converting the select query and running the make table query.

Create the select query
Note: If you already have a select query that produces the data that you need, go to the next steps.

1) On the Create tab, in the Query group, click Query Design.
2) In the Show Table dialog box, double-click the tables from which you want to retrieve data. Each table appears as a window in the upper section of the query designer. Click Close when have finished adding the tables.
3) In each table, double-click the field or fields that you want to use in your query. Each field appears in a blank cell in the Field row of the design grid.
4) Optionally, add any expressions to the Field row.
5) Optionally, add any criteria to the Criteria row of the design grid.
6) Click Run to run the query and display the results in a datasheet.
7) Optionally, change your fields, expressions, or criteria and rerun the query until it returns the data that you want to place in your new table.

Convert the select query
1) Open your select query in Design view, or switch to Design view.
2) On the Design tab, in the Query Type group, click Make Table.
   a) The Make Table dialog box appears.
3) In the Table Name box, enter a name for the new table.
   -or-
   Click the down-arrow and select an existing table name.
4) Do one of the following:
   a) Place the new table in the current database:
      • If it isn’t already selected, click Current Database, and then click OK.
      • Click Run, and then click Yes to confirm the operation.
      Note: If you are replacing an existing table, Access first deletes that table and asks you to confirm the deletion. Click Yes, and then click Yes again to create the new table.
b) Place the new table in another database:
   - Click Another Database.
   - In the File Name box, enter the location and file name of the other database.
     -or-
     Click Browse, use the new Make Table dialog box to locate the other database, and click OK.
   - Click OK to close the first Make Table dialog box.
   - Click Run and then click Yes to confirm the operation.
   Note: If you replace an existing table, Access first deletes that table and asks you to confirm the deletion. Click Yes, and then click Yes again to create the new table.

Delete data from an Access database by using a query
To delete large amounts of data quickly, or to regularly perform the same delete operations, consider using a query. A query can help you make sure that you are deleting the correct data, and can be saved and then easily repeated.

Note: If you want to remove a small number of records — any quantity that you feel comfortable deleting by hand — you can open the table in Datasheet view, select the fields or rows that you want to delete, and press DELETE.

Important: Make sure that you have a backup of your database before you delete the data.

Choose a query type based on the data you want to delete
You have the option of using either an update query or a delete query depending on the type of deletion that you need to perform.

Delete query
Use a delete query to remove entire records (rows) from a table or from two related tables, in one operation. Delete queries remove all the data in each field, including the key value that makes a record unique.

Note: If the records that you want to delete reside on the "one" side of a one-to-many relationship, you might need to change the relationship before you run the delete query.

Update query
Use an update query to delete individual field values from a table. An update query let you delete values by updating the existing values to either a null value (that is, no data) or a zero-length string (a pair of double quotation marks with no space between them).

What to verify before using a query to delete data
   - Ensure that the database file is not read-only. To do so, in Windows Explorer, right-click the database file and then click Properties.
   - Verify that you have the necessary permissions to delete records from the database. If you are not sure, contact your system administrator or the database designer.
   - Make sure that you have enabled content in the database. By default, Access blocks all action queries (delete, update, and make-table queries) unless you first trust the database.
   - Ask other users of the database to close all tables, forms, queries, and reports that use the data that you want to delete. This helps avoid lock violations.
Before you edit or delete records, back up the database. You cannot reverse operations that are performed by delete and update queries, so making a backup copy ensures that you can always reverse your changes.

**Tip:** If a large number of users connect to the database, you might need to close the database and then reopen it in *Exclusive* mode.

**To open a database in Exclusive mode**

1. Click the **File** tab, and then click **Open**.
2. Browse to and point to select the database, click the arrow next to the **Open** button, and then click **Open Exclusive**.

---

### Create and use a delete query

1. On the **Create** tab, in the **Queries** group, click **Query Design**.
   
   a) The **Show Table** dialog box appears.
2. Double-click each table from which you want to delete records, and then click **Close**.
   
   a) The table appears as a window in the upper section of the query design grid. The window lists all of the fields in the selected table.
3. Double-click the asterisk (*) to add all of the fields in the table to the design grid.
4. Optionally, double-click any field that you want to use to specify criteria for deletion, enter one or more criteria in the **Criteria** row of the designer, and then clear the **Show** check box for each criteria field.

   For example, suppose that a customer goes out of business and you need to remove all of the pending orders for that customer. To find just those records, you add the Customer ID and Order Date fields to the design grid, and then you enter the ID number of the customer that went out of business, and the date on which that customer's orders became invalid.

   **Important:** Use criteria to return only the records that you want to delete. Otherwise, the delete query removes every record in the table.
5. On the **Design** tab, in the **Results** group, click **View**, and then click the **Datasheet View**.
6. Verify that the query returns the records that you want to delete, and then press CTRL+S to save the query.
7. To run the query, double-click the query in the Navigation Pane.

### Deleting data from related tables

If you want to delete data from several related tables, you must enable the **Referential Integrity** and **Cascade Delete Related Records** options for each relationship. This allows your query to delete data from the tables on the "one" and "many" sides of the relationship. Preparing to delete related data requires verification of the following:

- Determine which records reside on the "one" side of the relationship and which reside on the "many" side.
- If you need to delete records on the "one" side of the relationship and the related records on the "many" side, you enable a set of rules called Referential Integrity, and you enable cascading deletes. Steps in this section explain Referential Integrity, and how to perform both tasks.
• If you need to delete records only on the "one" side of the relationship, you first delete that relationship, and then delete the data.

- or -

If you need to remove data only on the "many" side of the relationship, you can create and run your delete query without having to change the relationship.

**Determine the relationships**

To determine which records reside on the "one" and "many" sides of a relationship:

• On the **Database Tools** tab, in the **Relationships** group, click **Relationships**.

The **Relationships** tab appears and displays all of the tables in your database and also the relationships between each table and every other table. Each relationship is depicted as a line that connects tables between fields.

The figure shows a typical relationship. Most, if not all, of the relationships in a database have a "one" side and a "many" side. The relationship diagram denotes the "one" side with the numeral one (1) and the "many" side with the infinity (∞) symbol.

As a rule, when you delete records on the "one" side of the relationship, you also delete all of the related records on the "many" side of the relationship. However, when you delete records on the "many" side of a relationship, you typically don't delete records on the "one" side.

Also, by default, Access enforces a set of rules called referential integrity. These rules ensure that the foreign keys in a database contain the correct values. A foreign key is a column whose values match the values in the primary key column of another table.

**Edit the relationship**

Follow these steps only when you need to delete data on the "one" and "many" sides of a relationship.

1) On the **Database Tools** tab, in the **Relationships** group, click **Relationships**.

2) Right-click the relationship (the line) connecting the tables involved in the deletion operation, and then click **Edit Relationship** on the shortcut menu.

   a) The **Edit Relationships** dialog box appears.

3) Ensure that the **Enforce Referential Integrity** check box is selected.

4) Select the **Cascade Delete Related Records** check box.

   **Note:** Until you disable this property again, deleting a record on the "one" side of the relationship will delete all of the related records on the "many" side of the relationship.

5) Click **OK**, close the **Relationships** pane, and then go on to the next set of steps.
Delete a relationship
1) If you haven't done so already, open the Relationships pane.
   a) Make a note of the fields involved in the relationship so that you can restore the relationship after you delete your data.
3) Right-click the relationship (the line) connecting the tables involved in the deletion operation, and then click Delete on the shortcut menu.

Note: To restore the relationship, follow the previous steps to open the Relationships pane, and then drag the primary key field from the "one" table and drop it on the foreign key field of the "many" table. The Edit Relationship dialog box appears. If the old relationship enforced referential integrity, select Enforce Referential Integrity, and then click Create. Otherwise, just click Create.

Why am I seeing this error message, and how do I fix it?
If you build a delete query by using multiple tables and the query's Unique Records property is set to No, Access displays the error message Could not delete from the specified tables when you run the query.

To fix this problem, set the query's Unique Records property to Yes.

1) Open the delete query in Design view.
2) If the query property sheet is not open, press F4 to open it.
3) Click the query designer to show the query properties (rather than the field properties).
4) In the query property sheet, locate the Unique Records property, and set it to Yes.

Additional information

Back up the database
1) Click the File tab, point to Save & Publish, and then, under Save Database As, click Back Up Database.
   a) Access closes the original file, creates a backup, and then reopens the original file.
2) In the Save As dialog box, specify a name and location for the backup copy, and then click Save.
3) To revert to a backup, close and rename the original file so that the backup copy can use the name of the original version. Assign the name of the original version to the backup copy, and open the renamed backup copy in Access.

Stop Disabled Mode from blocking a query
By default, if you open a database that you have not chosen to trust or that does not reside in a trusted location, Access blocks all action queries from running.

If you try to run an action query and it seems like nothing happens, check the Access status bar for the following message:

This action or event has been blocked by Disabled Mode.

When you see that message, take the following step to enable the blocked content:

- On the Security Warning Message Bar, click Enable Content, and run your query again.
<table>
<thead>
<tr>
<th>Criteria</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; 234</td>
<td>Returns all numbers greater than 234. To find all numbers less than 234, use &lt; 234.</td>
</tr>
<tr>
<td>&gt;= &quot;Callahan&quot;</td>
<td>Returns all records from Callahan through the end of the alphabet.</td>
</tr>
<tr>
<td>Between #2/2/2007# And #12/1/2007#</td>
<td>Returns dates from 2-Feb-07 through 1-Dec-07 (ANSI-89). If your database uses the ANSI-92 wildcard characters, use single quotation marks (') instead of pound signs. Example: Between '2/2/2007' And '12/1/2007'.</td>
</tr>
<tr>
<td>Not &quot;Germany&quot;</td>
<td>Finds all records where the exact contents of the field are not exactly equal to &quot;Germany.&quot; The criterion will return records that contain characters in addition to &quot;Germany,&quot; such as &quot;Germany (euro)&quot; or &quot;Europe (Germany)&quot;.</td>
</tr>
<tr>
<td>Not &quot;T*&quot;</td>
<td>Finds all records except those beginning with T. If your database uses the ANSI-92 wildcard character set, use the percent sign (%) instead of the asterisk (*).</td>
</tr>
<tr>
<td>Not &quot;*t&quot;</td>
<td>Finds all records that do not end with t. If your database uses the ANSI-92 wildcard character set, use the percent sign instead of the asterisk.</td>
</tr>
<tr>
<td>In(Canada,UK)</td>
<td>In a list, finds all records containing Canada or UK.</td>
</tr>
<tr>
<td>Like &quot;[A-D]&quot;*&quot;</td>
<td>In a Text field, finds all records that start with the letters A through D. If your database uses the ANSI-92 wildcard character set, use the percent sign instead of the asterisk.</td>
</tr>
<tr>
<td>Like &quot;+ar*&quot;</td>
<td>Finds all records that include the letter sequence &quot;ar&quot;. If your database uses the ANSI-92 wildcard character set, use the percent sign instead of the asterisk.</td>
</tr>
<tr>
<td>Like &quot;Maison Dewe?&quot;</td>
<td>Finds all records that begin with &quot;Maison&quot; and that also contain a 5-letter second string in which the first 4 letters are &quot;Dewe&quot; and the last letter is unknown (indicated by a question mark). If your database uses the ANSI-92 wildcard character set, use the underscore (_) instead of the question mark.</td>
</tr>
<tr>
<td>#2/2/2007#</td>
<td>Finds all records for February 2, 2007. If your database uses the ANSI-92 wildcard character set, surround the date with single quotation marks instead of pound signs (#). Example: '2/2/2007'.</td>
</tr>
<tr>
<td>&lt; Date() - 30</td>
<td>Returns all dates more than 30 days old.</td>
</tr>
<tr>
<td>Date()</td>
<td>Returns all records containing today's date.</td>
</tr>
<tr>
<td>Between Date() And DateAdd(&quot;M&quot;, 3, Date())</td>
<td>Returns all records between today's date and three months from today's date.</td>
</tr>
<tr>
<td>Is Null</td>
<td>Returns all records that contain a null (blank or undefined) value.</td>
</tr>
<tr>
<td>Is Not Null &quot;&quot;</td>
<td>Returns all records that contain a zero-length string. You use zero-length strings when you need to add a value to a required field, but you don't yet know what that value is. For example, a field may require a fax number, but some of your customers may not have fax machines. In that case, you enter a pair of double quotation marks with no space between them (&quot;&quot;’) instead of a number.</td>
</tr>
</tbody>
</table>