 <p>Academic Computing CENTER American University of Beirut http://staff.aub.edu.lb/~acc</p>	<p>Part of this tutorial was adapted from a tutorial by Florida Golf Coast University http://www.fgcu.edu/support/office2000/Access Bay City Schools http://www.bcschools.net/staff/AccessHelp.htm Jim Fuller Tutorial http://www.teachers.ash.org.au/jfuller/Access/Access2000.htm James Madison University http://www.jmu.edu/computing/tutorials/Microsoft/Access</p>
--	---

Access II - 2003 Tutorial

I- Tables and Relationships

Using Related Tables

Creating a Relationship

Setting Referential Integrity

Adding a Table in the Relationships Window

II- Queries

Using Queries and RecordSets

Using the Simple Query Wizard

Creating a Query in Design View

Opening a Query

Adding a Table to a Query

Running a Query

Sorting a Query

Adding Criteria to a Query

Using Comparison Operators

Using an AND Condition

Using an OR Condition

Using a Wildcard Character

Creating a Parameter Query

Creating a Calculated Field

Creating Aggregate/Function Query

Creating an Action Query

Using Multiple Tables in a Query

Tables and Relationships

Using Related Tables

Tables can be joined, or related, in order to **Access** and coordinate information in all the fields of the related tables. Joining tables is a useful way to avoid the need to enter duplicate information in various, related tables. In addition, it allows you to create reports, forms, and queries from the related data tables and save them in the database file. Relating tables allows you to create smaller, more efficient tables that can be related when you need **Access** to the data.

When you relate tables, the table from which you select the first join field is the primary table and the table to which you drag the join field is the related table. The tables must have some common fields that contain the same type of data. One of the fields in the first table must be the primary key so that **Access** does not allow duplicate entries. The common fields in both tables must have the same or equivalent data types and; if they are **Number** fields, they must have the same field size.

Customer Number	First Name	Last Name	Address
1	Fouad	Halwany	Ras Beirut
2	Ziad	Farah	Baabda
3	Joe	Naim	Jounieh

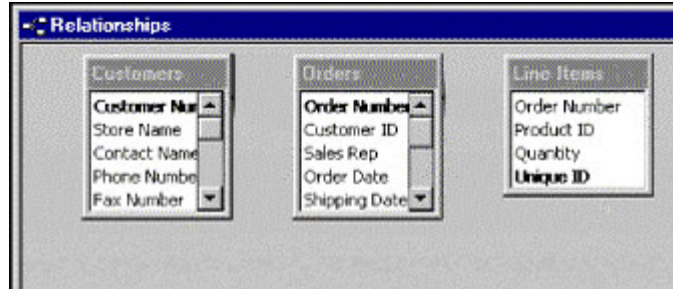
Customer Number	Item	Quantity
1	Fridge	1
1	Carpet	2
1	Computer	4
2	Chair	7
2	Table	3
3	Table	2

For example, you can create a table consisting of customer names, addresses, and telephone numbers. You can also include a unique identification number for each customer, which would be the primary key in the table. You can create this number or allow **Access** to create it for you. You could then create a separate table consisting only of orders placed by customers. This table would also contain the field for the unique customer identification number, but not the customers' names, addresses, and telephone numbers. By relating the two tables through the common customer identification number field, the customers' names, addresses, and telephone numbers do not have to be entered for every order.

Access includes two basic types of relationships: one-to-many and one-to-one. A one-to-many relationship occurs when one record from the primary table matches many records from the related table; for example, one customer record matches many order records. A one-to-one relationship occurs when one record from the primary table matches one record from the related table. **Access** determines the relationship type automatically when you create the relationship.

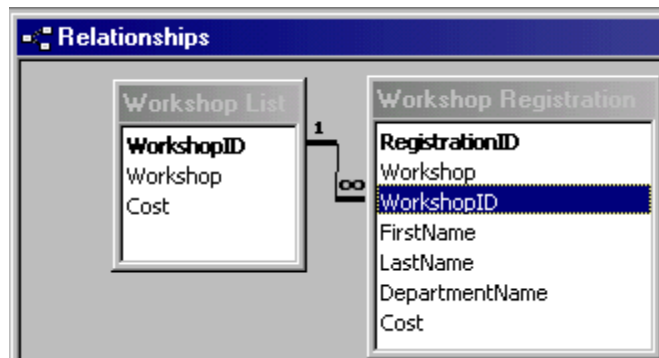
Adding a Table in the Relationships Window

You can add additional tables to the **Relationships** window. For example, if you have two related tables and then decide you need to *Access* information from a third table, you can easily add the required table to the **Relationships** window. Click on the **show table** button from the **shortcut** toolbar, select the table that you want to add and click on **Add** button. When done click on **Close** button.



Creating a Relationship

You create relationships between tables or queries in the **Relationships** window. The **Relationships** window displays a graphic representation of the relationships in the database. The field name representing the **primary key** in every table appears in bold.



Note: All tables must be closed before you can create relationships.

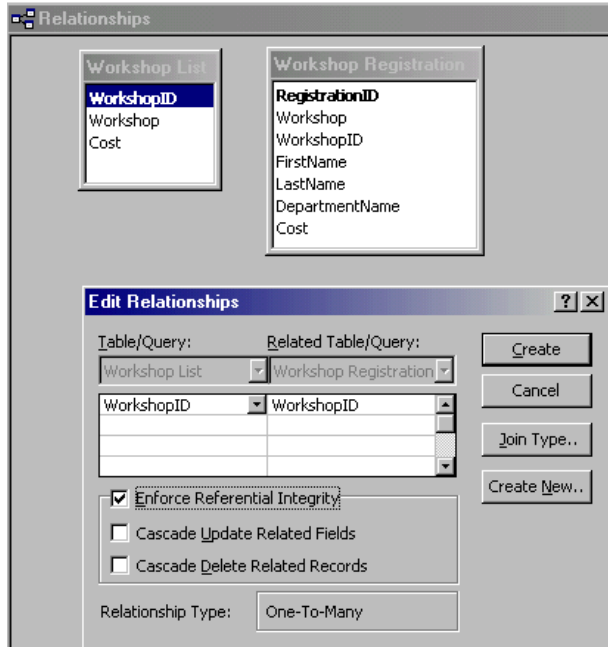
Setting Referential Integrity

When you create a relationship between two tables, you can set **Referential integrity**. **Referential integrity** is a built-in set of rules *Access* uses to make sure that the relationship is valid. **Referential integrity** can also prevent accidental deletion or editing of data. In order to use **Referential integrity**, the following conditions must be true: the related field in the primary table is the **primary key**, the related fields in both tables have the same data type, and both tables belong to the same database.

When you set **Referential integrity**, you must observe the following three rules. First, you cannot enter data in the join field in the related table that does not have a match in the join field in the primary table. Second, you cannot delete records from the primary table if there are matching records in the related table. Third, you cannot edit primary key values in the primary table if related records exist. However, if you want to perform the changes listed above and still maintain referential integrity, you can select the **Cascade Update Related Fields** and **Cascade Delete Related Records** options in the **Edit Relationships** dialog box. When either or both of these options are selected, *Access*

makes the necessary changes to the related tables automatically to maintain **Referential integrity**. It is recommended that these two options be used after careful consideration since the changes cannot be undone.

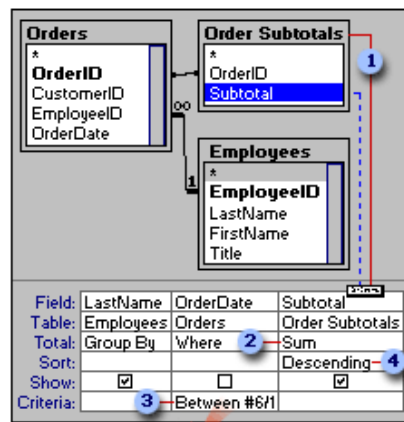
When the **Referential integrity** option is activated, *Access* displays symbols above the join line in the **Relationships** window to indicate the type of relationship: **one-to-one** or **one-to-many**. The number 1 above the join line next to a field list indicates "one", while the mathematical symbol for infinity (which resembles a horizontal 8) indicates "many".



Queries

Using Queries and RecordSets

A query is a question about information in one or more tables. You can use queries to view and analyze data or as a basis for forms and reports. Queries are commonly used to display fields from related tables and enable you to control not only what records display, but also what fields display. For example, you may want a list of the contacts and telephone numbers for a particular region to give to one of your sales representatives. By creating what is called a **Select** query, you can limit the records to the appropriate region and limit the fields to the contact name and telephone number.



Zoom
Between #6/1/01# And #6/15/01#

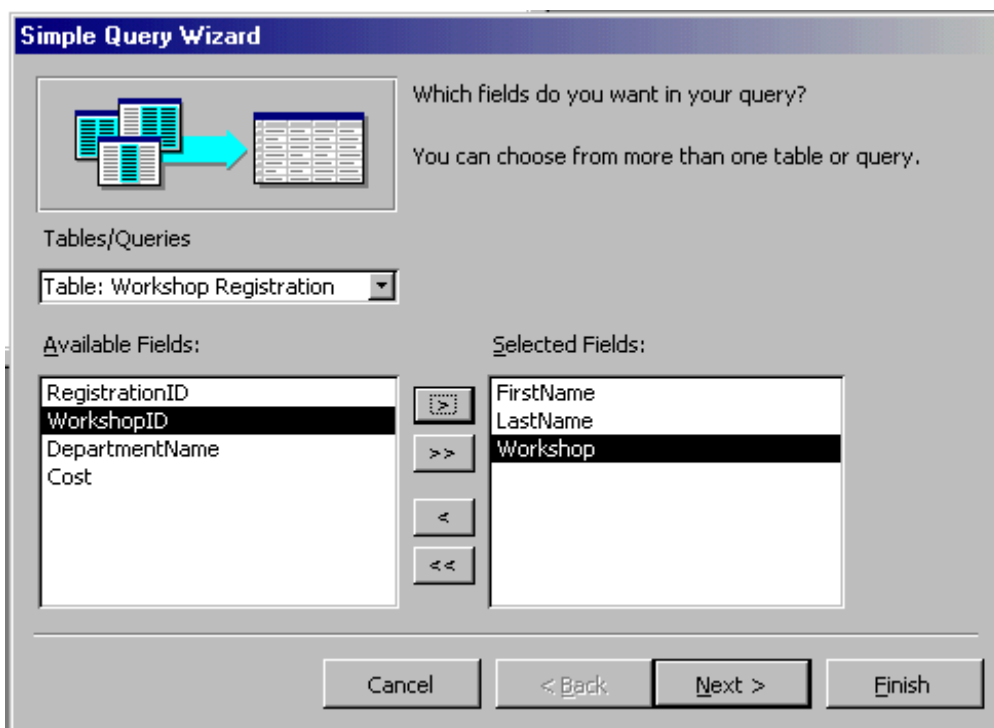
- 1 Add or remove tables, queries, and fields
- 2 Calculate amounts
- 3 Limit results using criteria
- 4 Sort records

A query does not contain data. Rather, it is a set of instructions. **Access** uses these instructions to select and display the appropriate records in a table. Therefore, when you add data to a table, you do not have to update the query. The query always considers all the data in the table. If the new records meet the conditions of the query, they will be included when the query results appear.

When you open or run a query, a RecordSet appears. A RecordSet contains all the fields and records that meet the conditions of the query. While the RecordSet is not a table, it can be used under certain conditions to add and edit records in tables.

Using the Simple Query Wizard

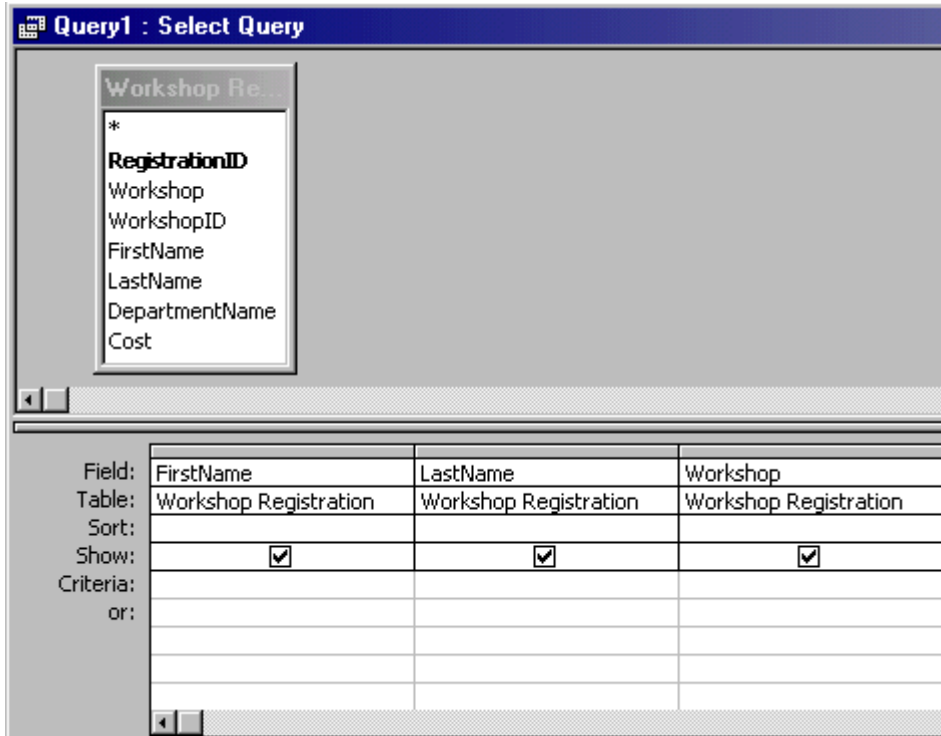
Access provides a Simple Query Wizard that guides you through the steps to create a basic select query. When you use the Simple Query Wizard, you select the table you want to use and the fields you want to display in the query. In the last step, you name the query and then choose whether or not to display the results (the RecordSet) of the query or if you want to go to **Design View** to change the design of the query.



Creating a Query in Design View

You can create a query in **Design View**. This option gives you the most flexibility in designing a select query. It allows you to add criteria to select records and sort the resulting RecordSet.

When you create a query in **Design View**, you use the design grid to set up the query. The field list of the table you want to use in the query appears in the top pane of **Design View**. You add the fields you want to use in the query to the design grid in the bottom pane of **Design View**, along with any sort orders or criteria for selecting records.



Opening a Query

When you open a query, *Access* runs the instructions in the query and displays the resulting RecordSet in **Datasheet View**. If you have added records since the last time you ran a particular query, the new records will appear if they meet the query criteria.

Adding a Table to a Query

You can use more than one table in a query. The tables must be joined in order for the query to give accurate results. If they are not joined, you can create a join in the top pane of **Design View**.

When you add more than one table, the field lists appear in the top pane of **Design View**. If the tables are already related, join lines appear automatically.

Once you have added a table to the query, you can then add fields from the field list to the design grid. The second row in the design grid is the Table row, which indicates from which table the field originates.

When you open **Design View** to design a new query, the **Show Table** dialog box opens automatically so that you can add multiple tables. However, when you open **Design View** to modify an existing query design (i.e., to add a table), you must open the **Show Table** dialog box manually.

Running a Query

You can run a query directly from **Design View** to display the RecordSet. This option is useful if you want to test the design of the query to see if the resulting RecordSet contains the information you need.

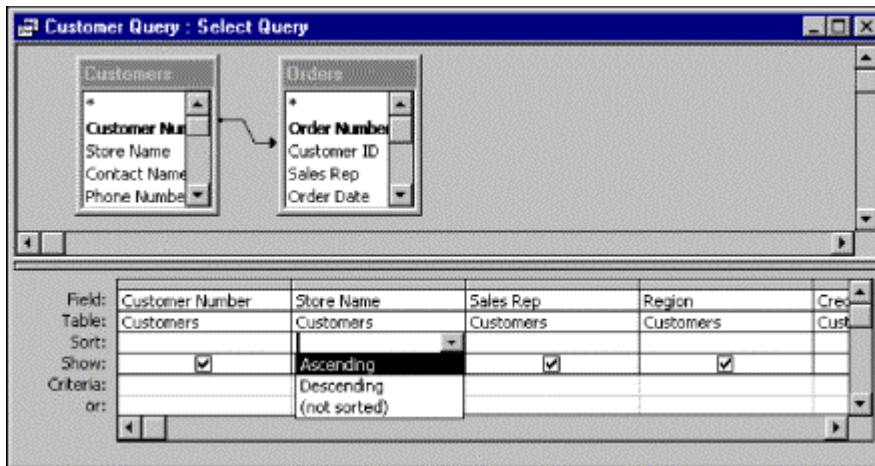
Running a query does not save the design of the query. If you close the RecordSet after running a query, a message box opens, asking if you want to save the changes.



Run Query button

Sorting a Query

When you run a query, the records in the RecordSet appear in the same order in which they appear in the table. You can sort the records by either sorting the RecordSet or assigning a sort order in the query design. You can sort the RecordSet just as you would sort a table. However, you must do this every time you run the query. If you assign the sort order in the query design, **Access** performs the sort automatically each time you run the query.



Adding Criteria to a Query

You can enter data in the **Criteria** row of the query design grid to restrict the number of records that are returned in a query.

To select records that meet a single value, you enter the value in the **Criteria** row under the appropriate field. **Access** automatically inserts quotation marks (" ") around

alphanumeric entries and number symbols (#) around date entries. If the entry is numeric, the number appears without quotation marks. When you run the query, only those records with values that match the criteria appear in the RecordSet.

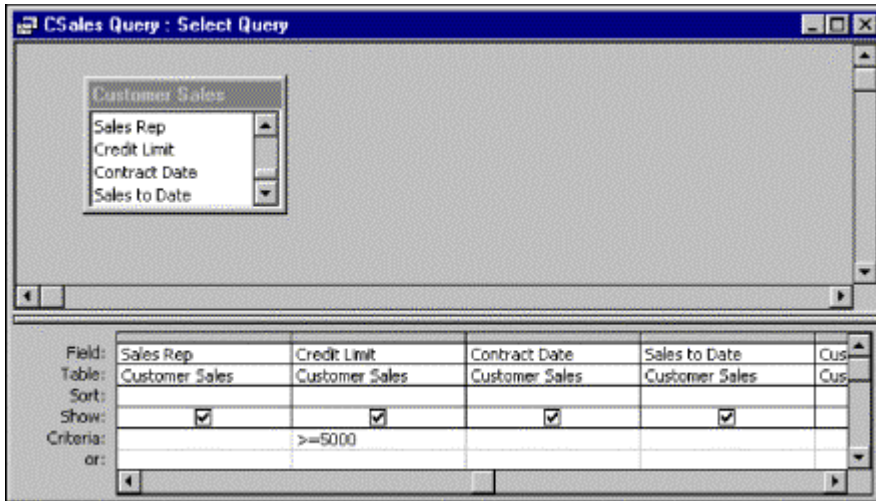
Using Comparison Operators

You can enter criteria in the **Criteria** row of the query design grid in order to select specific records. The simplest criteria requires that records match a single value to be included in the RecordSet.

You can also use comparison operators to select a specific group of records in a table. For example, if you want to find all customers with credit limits less than \$1000, or all customers with a contract date on or before January 1997, you can write an expression that defines the criteria using a combination of comparison operators and field values, such as <1000 or <=1/1/97. Comparison operators are symbols that represent conditions recognized by *Access*. The available comparison operators are

<i>Operator</i>	<i>Description</i>
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
=	equal to
<>	not equal to
Not	reverse logic

You can use a comparison operator to compare a specified value with all the values in a field. When you run the query, only the records with values meeting the criteria you specify appear in the RecordSet.



Using an AND Condition

Many times, a query requires more than one condition to obtain the desired result. For example, if you want to find all customers in PA with sales to date over \$10,000, you would need two conditions: State=PA and Sales to Date>10000. The records must meet both conditions in order to be included in the RecordSet. To combine two criteria in this way, you use the **And** logical operator.

You can use the **And** operator in a single field or in different fields. In a single field, you can use the **And** operator to find records that fall into a range. For example, to find customers whose contract dates fall between 9/1/99 and 9/30/99, you type both criteria on a single line in the **Criteria** row under the appropriate field (i.e., >=9/1/99 **And** <=9/30/99 in the **Contract Date** field).

The **And** operator also allows you to impose conditions in two different fields. For example, to find customers in PA with sales to date over \$10,000, you type each criterion on a single line in the **Criteria** row under the appropriate fields (i.e., =PA in the **State/Province** field and >10000 in the **Sales to Date** field).

Field:	First Name	Last Name	DepartmentName	StateorProvince
Table:	University Employee	University Employee	University Employee	University Employee
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:			"Education"	"Maryland"
or:				

Using an OR Condition

Many times, a query requires more than one condition to obtain the desired result. For example, if you want to find all customers in PA or all customers with sales to date over \$10,000, you would need two conditions: State=PA as well as Sales to Date>10000. The records only need to meet one of the conditions in order to be included in the RecordSet. To combine two criteria in this way, you use the **Or** logical operator.

You can use the **Or** operator in a single field or in different fields. In a single field, you type the criteria on two separate lines under the same field. In different fields, you type the criteria on two separate lines under the appropriate fields. For example, to find all customers with contract dates on or before 1/1/99 or credit limits above \$3,000, you type <=1/1/99 in the **Criteria** row under the **Contract Date** field and >3000 in the or row under the **Credit Limit** field.

Field:	First Name	Last Name	DepartmentName
Table:	University Employee	University Employee	University Employee
Sort:			
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:			"Psychology"
or:			"Music"

Using a Wildcard Character

Wildcard characters are used in a query to find records when the criteria contains a pattern (such as all last names beginning with M) or are only partly known (such as the proper spelling - Kline or Klein). Wildcards take the place of one or several letters in a Text field or numbers in a Date/Time field.

The two most common wildcards are listed in the following table:

<i>Wildcard - Representing</i>	<i>Example</i>
? - Any single letter or number	Sm?th finds Smith and Smyth, whereas ?andy finds Sandy, Randy, etc.
* - One or more letters or numbers	M* finds all records that start with M; 8/*/99 finds all dates in August, 1999; and *ball* finds all records that have the word ball anywhere in the field

Creating a Parameter Query

You can automate the process of changing the criteria for queries that you run on a regular basis by using parameter query. A parameter query prompts the user for a value to use it as a criteria field every time it's run thus eliminating the need to open the query in **Design View** to change the criteria manually.

To create a parameter query, under the criteria field type the prompt message between []. For example under a field called **Course Name** you may type [Enter Course Name:]. When you run the query you will get a prompt similar to the one below.



You may enter many parameters in your query with different functions. For example for a date field you might type [Start Date] And [End Date]. This will prompt you twice for a date then display the records that have a date value between the two entered above.

By default you are prompted for the parameter values as they are arranged from left to right in **Design View** . However you can override such an order by going to **Query > Parameters**. A small window will open where you can type in prompt messages between [] and specify the value data type. You should make sure that the prompt messages here are the same as those entered under the criteria fields.

Creating a Calculated Field

You may want to use field values in a table to calculate new values, such as multiplying the value in the **Participants** field by the value in the **Cost** field to calculate the total cost. **Access** allows you to use expressions to calculate new field values. The expression can also include a single field, which has a value that needs to be adjusted. In expressions, field names are enclosed in square brackets ([]); numbers are not.

For example, to calculate a new field Total Cost and display the results in a column you want to name **Total Cost**, you would enter **Total Cost:[Participants]*[Cost]** in the design grid. The colon is used to separate the column name from the expression.

You create calculated fields in queries. You can also use criteria to remove nonessential records, thereby allowing the query to run faster. The results can then be used to generate a report.

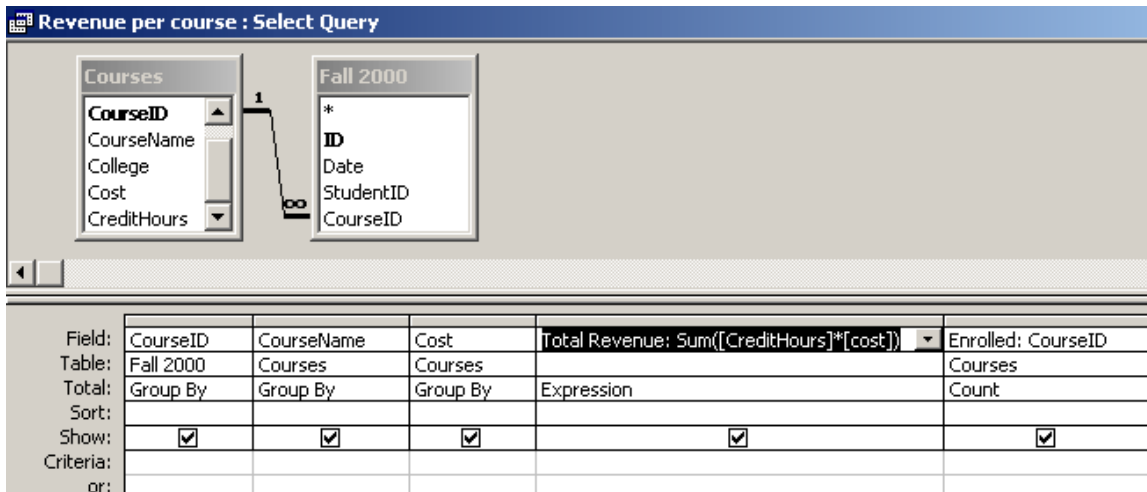
Field:	Workshop	Participants	Cost	Total Cost: [Participants]*[Cost] ▾
Table:	Calculate Total Cost	Calculate Total Cost	Calculate Total Cost	
Sort:				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:				
or:				


Creating Aggregate/Function Query

Access allows you to create a query that groups records by a selected field and applies a function that calculates values on other fields in the query according to your needs. For example, you can group records in a table by state and then select the **Count** function to find out how many customers (records) are in each state (field). You can also group by customer name (field) and calculate the **Sum** of each customer's orders (record values).

There are several types of functions from which you can choose. The most commonly used functions are listed in the following table:

<i>Function</i>	<i>Description</i>
Sum	Sums the values in the calculated field
Average	Finds the average value of the calculated field
Count	Counts the number of records in the calculated field
Max	Finds the highest value in the calculated field
Min	Finds the lowest value in the calculated field



To create such query, you have to **click** the **Totals** button on the Database toolbar  in **Design View**. A new row will be added to your work area where you can select on of the functions mentioned above to **Group By** it.

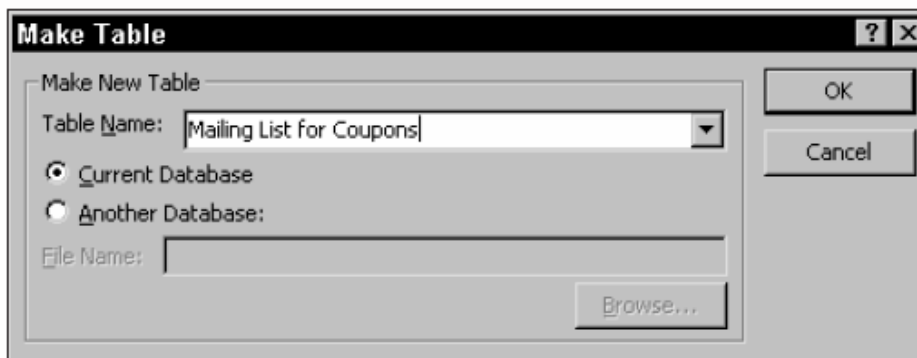
Creating an Action Query

Action queries are queries that do something more than simply select a specific group of records and present them to the user. When you create a query *Access* automatically creates it as a select query. However you can specify a different type from the Query design menu. The menu's selections are: Make Table, Update, Append, and Delete.

Update Queries: They are used if you want to modify a field in a specific table. Instead of going through the entries and changing them one by one an update query will help you do that in a faster and more reliable way. From the **Query** menu in the **Main Menu** bar select **Update Query** select the **Update to** cell under the field that you want to change type in the new value and run the **Query**.

Field:	City	Type of Animal	Current Vaccinations		
Table:	Customer	Pets	Pets		
Update To:			Yes		
Criteria:	"Mountain View"	"Horse"			
or:					

Make-table Queries: They are used when you apply certain criteria options to your select query and you want these result to be saved in a table. Simply convert your query to make-table one from **Query > Make-Table**, after you have selected the criteria. Enter the table new name under table name and run the query that will create a new table from the results.

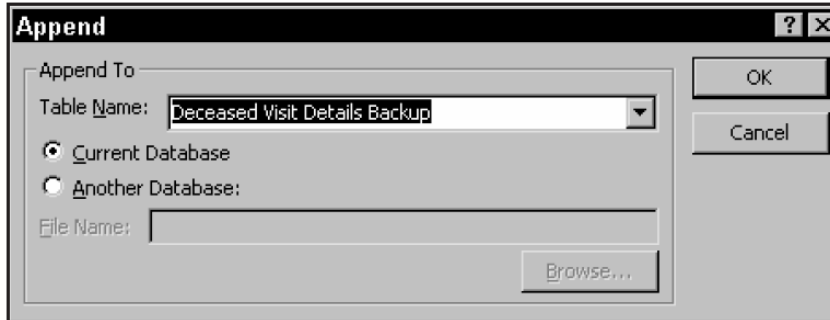


Append Queries: Allow you to move records from one table to another directly instead of the copying and pasting process. However you should be aware of some issues:

- If the table you are adding record to, has a primary key field, the records you add can not have Null or duplicate primary key values.

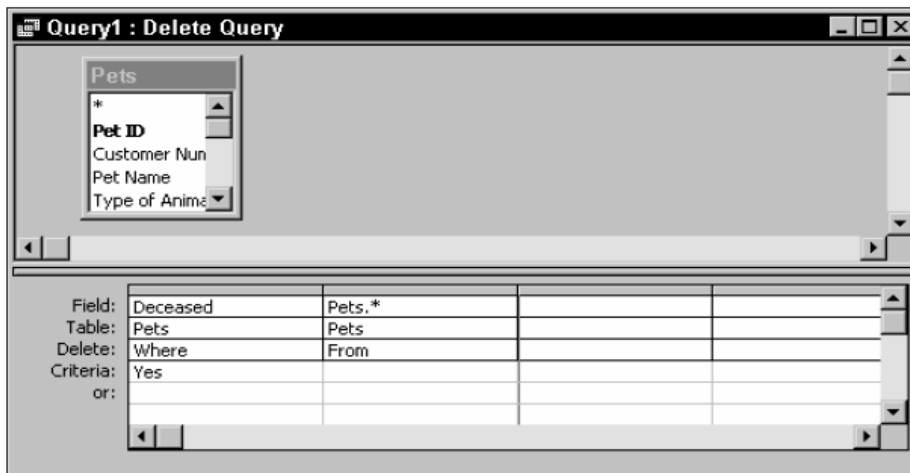
- If you add records to another database you should know the name and location of the database.
- If you append records with Auto Number field, do not include it if the table you are adding to already have one. This might cause duplicate values and confusing results.

In order to append your data to a previous table convert you query to an **Append Query** using **Query > Append Query**. Under **Append to** choose the table you want to append the data to and run the query. The new data will be added to your table.



Delete Queries: Allow you to delete certain records from a table directly instead of going through the table’s entries one by one. For example you might wish to delete all students whose IDs start with 2000 from your table. A **Delete Query** allows you to do that simply by running it after entering the criteria options.

In order to use the above method you should convert your query type to a **Delete Query** from **Query > Delete Query**, and run the query after entering the criteria.



Using Multiple Tables in a Query

There may be instances when you need to add more than one table or query to a query. In order to do this, you need to ensure that the field lists are joined to each other with a join line so that *Access* knows how to connect the information.

If tables in a query are not joined to one another, *Access* does not know which records are associated with which, so every combination of records between the two tables would appear in the query. Therefore, if each table had 50 records, the query's results would contain 2500 records (50x50), thereby rendering useless results.

If you previously created a relationship between tables in the **Relationship** window, *Access* automatically displays join lines when you add related tables in query **Design View**. If you have not previously created relationships, *Access* automatically creates joins if tables were added to a query as long as the tables each have a field with the same or compatible data type and one of the join fields is a primary key. If tables added to the query do not include any fields that can be joined, you have to add one or more extra tables or queries to serve solely as a bridge between the tables containing the data you want to use.

.....