Database Fundamentals

Climsoft Version 3.0 Training Manual

March 2011

Contents

Introduction

By definition, a database is an organized collection of related data items in digital form. Although data can be organized to a certain extent using an application like a spreadsheet, the level of organization in the case of a database is much higher, and is achieved by following the principles and rules of database design. A software application designed for working with databases is called a database management system (DBMS). Currently, the most widely used type of DBMS is a relational database management system (RDBMS). The most common RDBMS designed for small databases is MS Access, which comes as part of MS Office. Other common RDBMS usually used for large or enterprise level databases are MS SQL Server, Oracle and the open source MySQL. The standard computer language underlying relational databases is called Structured Query Language.

Another type of database we encounter whenever we start using a PC running on the more recent Windows Operating Systems like Windows XP is the hierarchical database system called the Windows Registry. This is used to store all information about the software installed on the PC for example the folder locations of a particular application like MS Office, antivirus software, in fact any application stored on the PC. The information in the Registry is stored in a hierarchical fashion, the root of the information being the entire computer, which branches into hives, followed by keys, then values and data.

In relation to Climsoft, knowledge of how information is stored in the Registry is important in troubleshooting problems which may occur with the Climsoft installation.

Fig 1 is a snapshot of the Windows Registry showing the location where Apache web server program is installed on a particular computer.

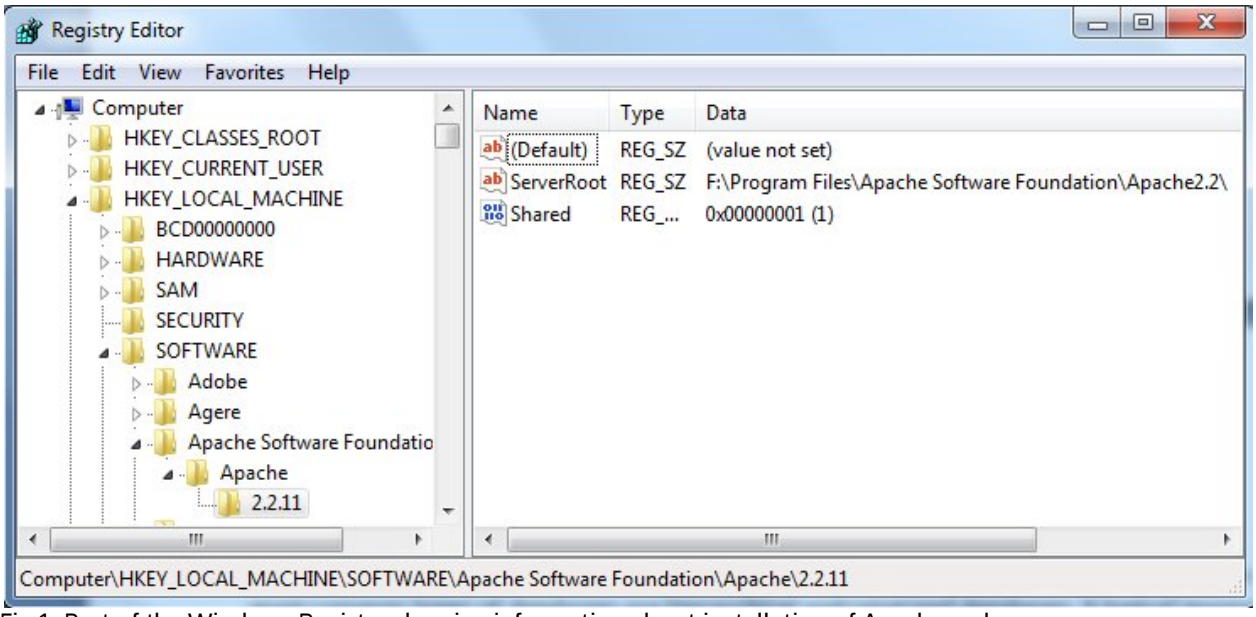


Fig 1. Part of the Windows Registry showing information about installation of Apache web server.

Limitations of a Spreadsheet in Climatological Data Management

A spreadsheet is designed to store data in a worksheet made up of rows and columns. The worksheet is the major object of a spreadsheet. Another object which can be part of a spreadsheet, but perhaps not so commonly used, is the macro object used for automating tasks in a spreadsheet.
Apparently the structure of a spreadsheet with rows and columns corresponds to the way data are stored on climatological returns. So at first glance, one would be inclined to use a spreadsheet to store climatological data since a spreadsheet seems to offer a direct visual mapping between the data on the paper form and the data on the computer.

After storing the data, a fundamental question is how to retrieve a subset of data requested or required for a particular purpose. For example the requested data may be daily maximum and minimum temperatures for specified stations and for a period like thirty years. In a typical spreadsheet one would be limited to a process of Copy and Paste. And in the first place, the selection of the data is visual. There is no provision to specify the criteria for the required data. The process would become more difficult if the data request is for the average number of rainy days (with threshold of say 0.3mm), for the month of April over a thirty year period.

In climatological data management, the quality of data is also critical. According to the WMO Guidelines to Climatological Practices, climatological data should go through a number of quality control (QC) checks. One of the QC checks is for internal consistency, for example to check that for a given station and observation time, the maximum temperature is greater or equal to wetbulb temperature. Other QC checks include spatial consistency i.e. comparison between values of the same element observed at the same time at neighbouring stations. Attempting to implement such QC checks in a spreadsheet would seriously expose the limitation of a spreadsheet in enforcing data quality in climatological data management.

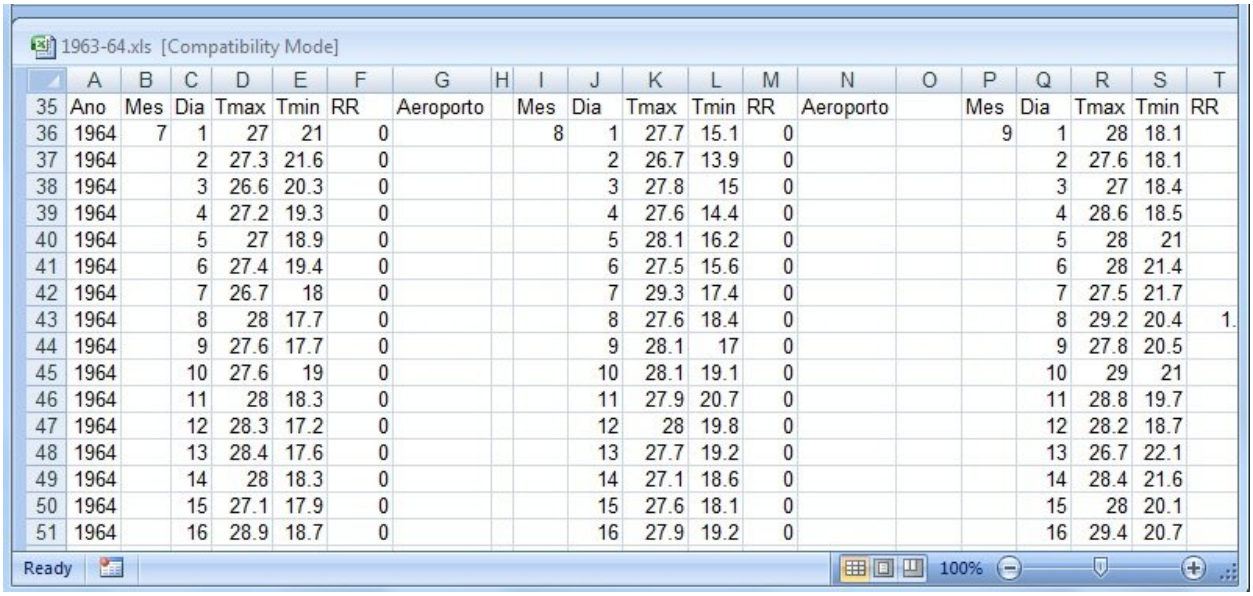
An essential component of climatological data is metadata i.e. data giving more information about observation data. A mere observation value would be meaningless if for example there is no information about where the observation reading was taken, the units of measurement, time of observation etc. Other important metadata, particularly in the analysis of Climate Change, is information on station movement over time. Change of instruments used for measuring a particular element in the history of a station is also essential.

Storing all such metadata in a way that obeys strict rules concerning how the metadata are related to an observation would be beyond the capability of a spreadsheet. An example of a rule to be obeyed would be that, for a particular station and a particular instrument, at a given time and observation level, there can only be one observation value for example wind direction.

Security features which can be enforced in a spreadsheet fall short of the type of security required for a professional climatological database management system. Different user roles and different access rights are required. For example some users can only be allowed to enter

data but are not allowed to retrieve the data. Some users would only have access only to a particular type of data e.g. metadata. This type of security is known as user-level security as is typically available in a proper DBMS.

Fig 2. Shows a screenshot of sample climatological data stored in a spreadsheet. In this particular example, daily data for a particular station and a particular station are stored in one worksheet. To put together data for different years would require opening different worksheets.


1963-64.xls [Compatibility Mode]

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
35	Ano	Mes	Dia	Tmax	Tmin	RR	Aeroporto		Mes	Dia	Tmax	Tmin	RR	Aeroporto		Mes	Dia	Tmax	Tmin	RR
36	1964	7	1	27	21	0			8	1	27.7	15.1	0			9	1	28	18.1	
37	1964		2	27.3	21.6	0				2	26.7	13.9	0				2	27.6	18.1	
38	1964		3	26.6	20.3	0				3	27.8	15	0				3	27	18.4	
39	1964		4	27.2	19.3	0				4	27.6	14.4	0				4	28.6	18.5	
40	1964		5	27	18.9	0				5	28.1	16.2	0				5	28	21	
41	1964		6	27.4	19.4	0				6	27.5	15.6	0				6	28	21.4	
42	1964		7	26.7	18	0				7	29.3	17.4	0				7	27.5	21.7	
43	1964		8	28	17.7	0				8	27.6	18.4	0				8	29.2	20.4	1.
44	1964		9	27.6	17.7	0				9	28.1	17	0				9	27.8	20.5	
45	1964		10	27.6	19	0				10	28.1	19.1	0				10	29	21	
46	1964		11	28	18.3	0				11	27.9	20.7	0				11	28.8	19.7	
47	1964		12	28.3	17.2	0				12	28	19.8	0				12	28.2	18.7	
48	1964		13	28.4	17.6	0				13	27.7	19.2	0				13	26.7	22.1	
49	1964		14	28	18.3	0				14	27.1	18.6	0				14	28.4	21.6	
50	1964		15	27.1	17.9	0				15	27.6	18.1	0				15	28	20.1	
51	1964		16	28.9	18.7	0				16	27.9	19.2	0				16	29.4	20.7	

Ready 100%

Fig 2. Sample climatological data stored in a spreadsheet.

Relational Database Design

Relational database management systems are based on relational mathematics. From a more practical point of view, the data are stored in rows and columns, similar to a spreadsheet. The database objects used for storing data are known as tables. In relational database terminology, the rows are referred to as records while the columns are known as fields. Unlike a spreadsheet, a database needs to be designed before any data can be stored. RDMBS applications like MS Access offer tools for designing databases, but the design can be produced independently of a RDBMS application.

In a properly designed database, a table should store only data which can be logically grouped together and identified as belonging to a particular entity like a station. Each entity must have a name and one or more attributes making up the entity. For example in meteorology, if we consider a station as an entity, it will have attributes like *station name*, *station ID*, *latitude*, *longitude*. See Fig 3. Another entity, shown in Fig 4, would be instrument which would have attributes like *instrument name*, serial number.

station
station_name
station_id
latitude
longitude

Fig 3. Station Entity

instrument
Instrument_name
serial_number

Fig 4. Instrument Entity

Rather than showing this design in graphical form we could also represent the design as follows:
Entity(attribute*1*, attribute*2*,…….attribute*n*). For example station(station name, station ID, latitude,longitude).

However there is need for a systematic approach to the identification of these entities. What should be identified first is the primary data we are interested in observing and storing. In meteorology or climatology, the primary data of interest is the observation. For an observation value like say 24.7 to be meaningful there would be need to know *what* (element) this value represents. A value of 24.7 could be temperature or precipitation. We also need to know *where* (station) the value was observed and *when* (datetime) the value was recorded. So at the first level of design, we would have observation as an entity as shown in Fig 5, with attributes of *station*, *element*, *datetime* , *observation level* and so on.

observation
station
Obs_element
Obs_datetime
Obs_level
Obs_value

Fig 5. Observation Entity

Exercise 1.
Draw up a list of as many attributes as possible for the entities station and instrument.

When we have drawn up an exhaustive list of attributes we then look at each attribute in turn to see if the attribute has its own attributes. For example if we look at *station* as an attribute of the observation entity, from a knowledge of meteorological practice, and as shown in previous sections, it would be observed that station has its own attributes like *station name*, *station ID*, latitude, *longitude*. In this case, we design separate secondary entities for such attributes like station, but still maintaining them as attributes of the original entity e.g. observation. This process of indentifying secondary entities from a primary entity is called normalization.

Exercise 2.
Looking at the entity observation in Fig 5, which other attribute of the observation entity has its own secondary attributes? Give a list of as many attributes of the secondary entity as you can find.

When giving names to database objects like entities and their attributes, it is essential to avoid use of reserved words, that is words that are used by the database management system e.g. value, element, level.

Table Design in MS Access

We now want to create a new database in MS Access and then design a new table.

Exercise 3.

Start MS Access. Click on the menu item File, then New to get the screen shown in Fig 6.

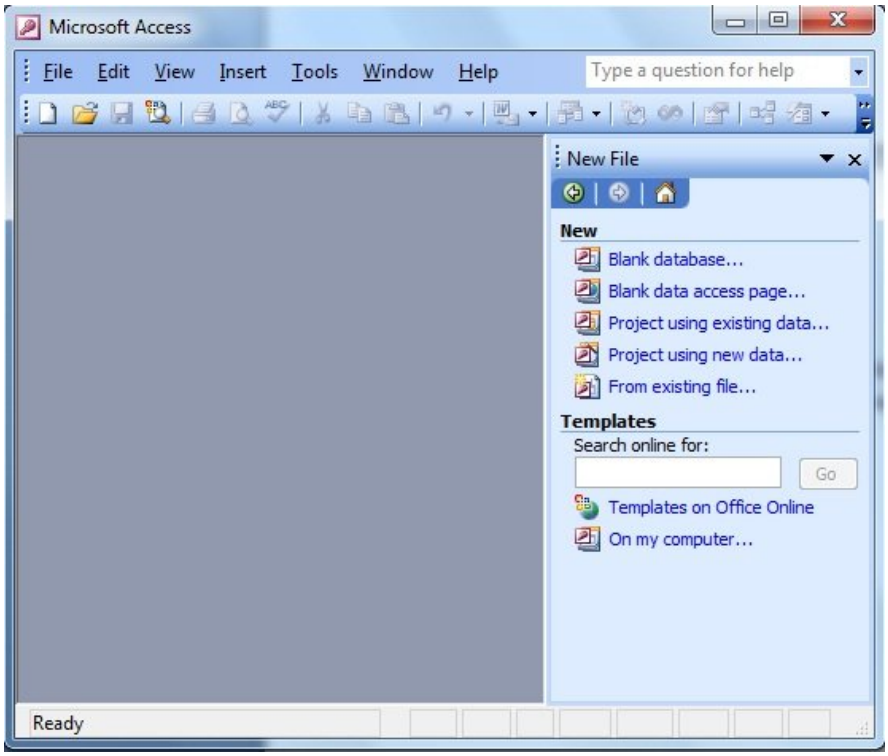


Fig 6. Creating new database in MS Access.

In the next step, click on Blank database to get the screen shown in Fig 7. Give the new database the name Climsoft_db1.mdb and save it in a new folder named Climsoft_training.

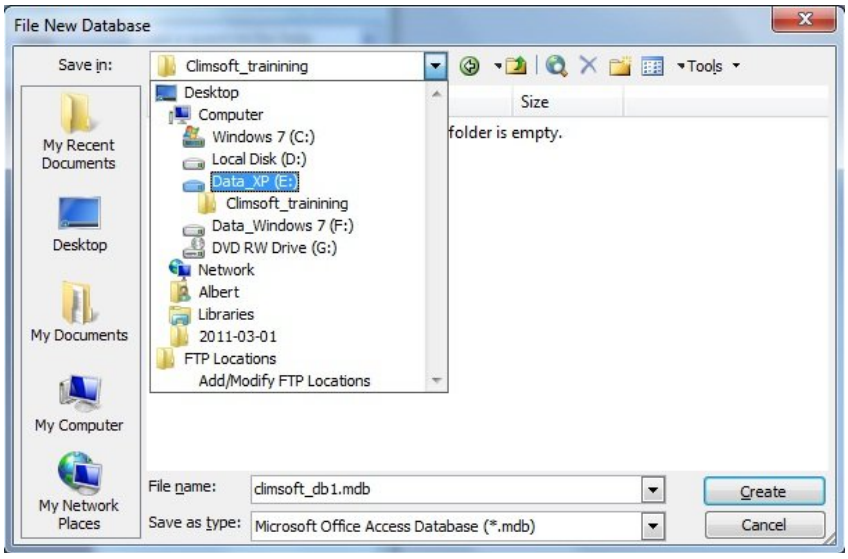


Fig 7. Saving a new database

After creating the new database you should get a dialogue similar to Fig 8, showing the types of database objects available in MS Access.

What are the different database objects in MS Access?

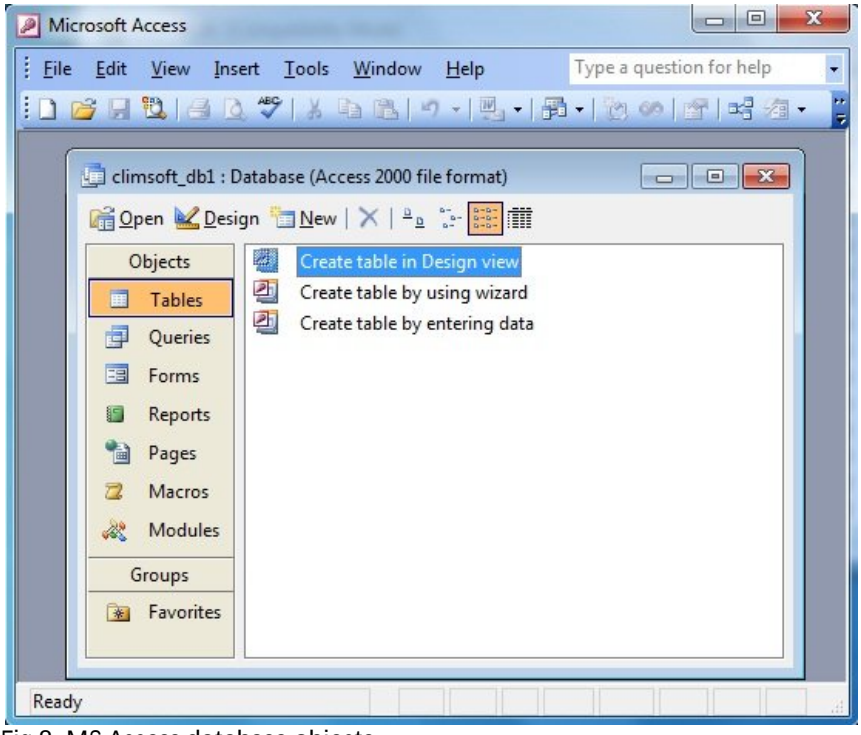


Fig 8. MS Access database objects.

Double click on Create table in Design view to get the new screen shown in Fig 9.

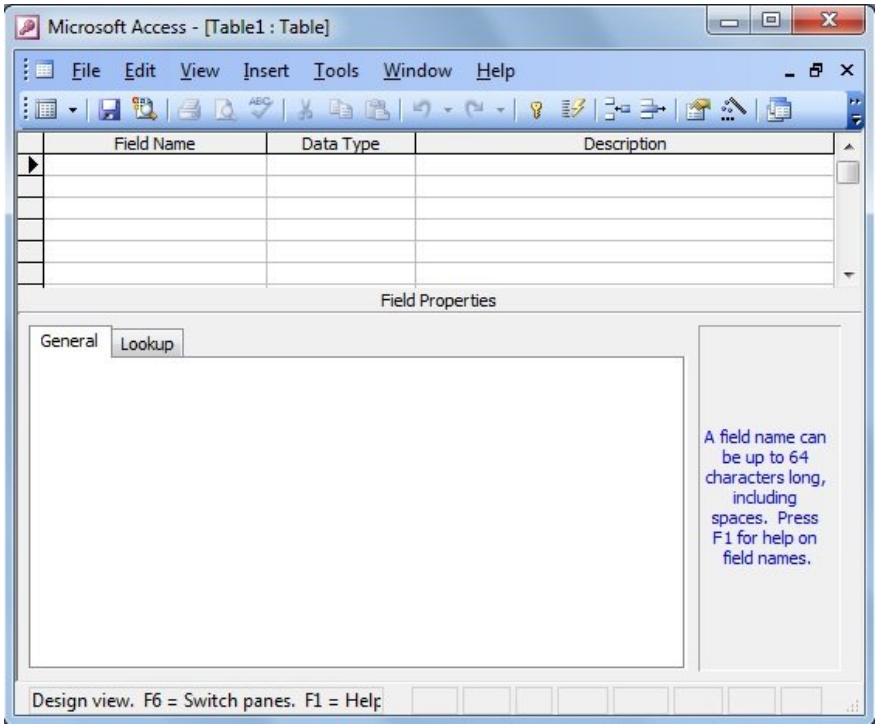


Fig 9. Table design window

Design a new table called station using information from Fig 3. Save your work without creating a Primary Key, by answering No to the dialogue which comes up asking you if you what to create a primary key.

You should then get the screen shown in Fig 10, with your newly created station.

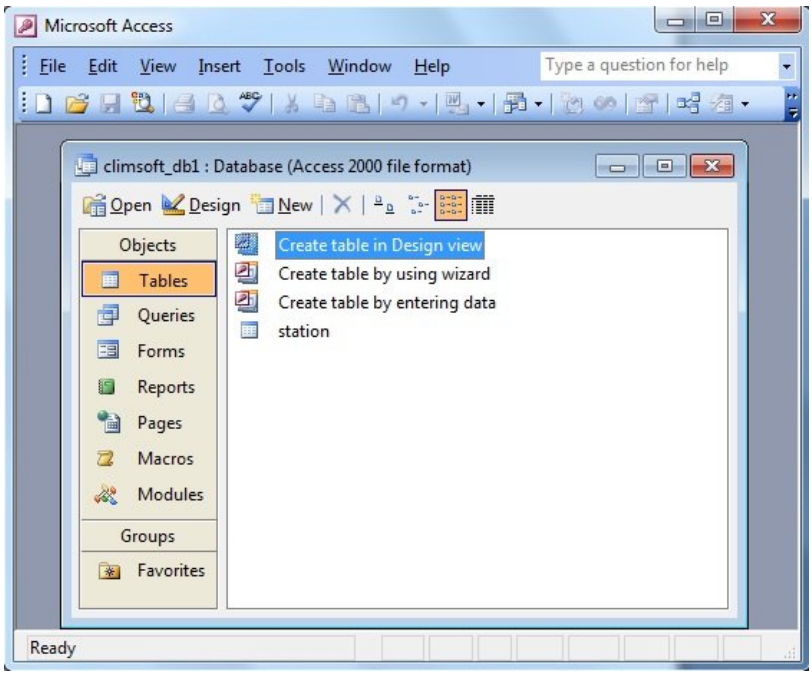


Fig 10. Newly created table

If you double click on the newly created table station, you will then get the new station object in datasheet view. This allows you to enter some data like you would do in a spreadsheet.

What difference do you notice between the table datasheet view and a spreadsheet ?

Field Data Types and Field Properties

A more detailed design of tables involves defining the data type for each field. In MS Access there are a number of data types which can be selected from a dropdown list as shown in Fig 11 below. The same figure also shows different field properties which can also be configured for each table field.

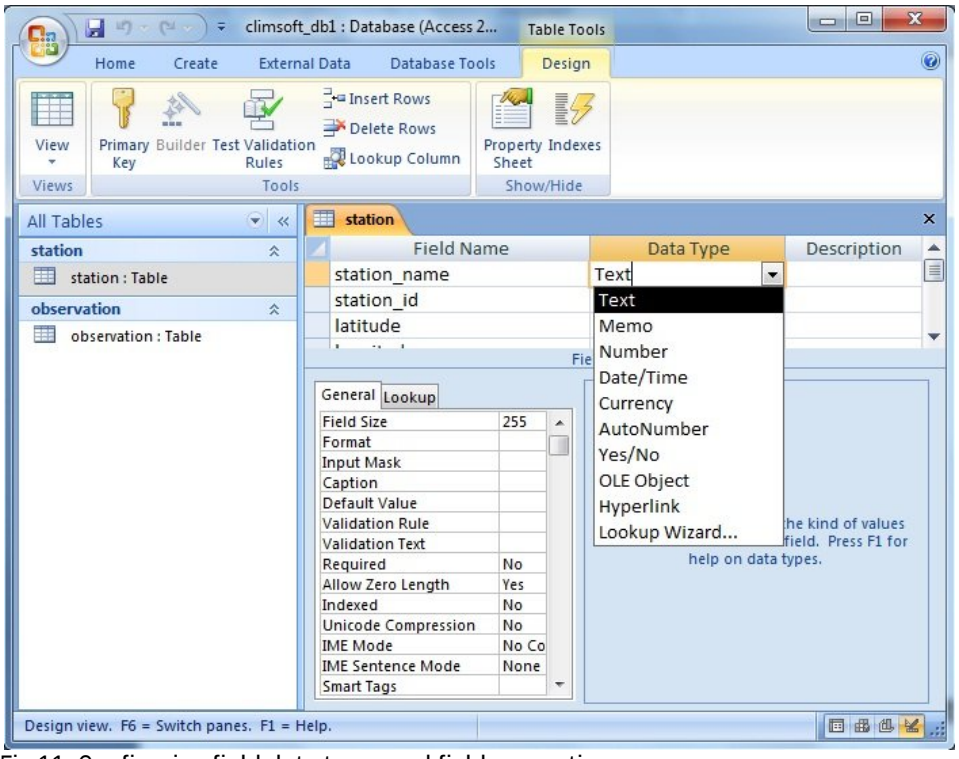


Fig 11. Configuring field data types and field properties

Primary Key and Index

A Primary Key is a field that uniquely identifies a table record and cannot contain a Null value. If a combination of fields is required to uniquely identify a record, then we talk of a compound or composite Primary Key. If a field or combination of fields is designed for record searching, such a field or combination of fields is termed an Index. An Index can also be defined as unique. A A field or combination of fields is configured as a Primary Key or Index by selecting the field or combination of fields and then clicking on the Primary Key icon or Indexes icon on the tools below the menu items in table design.

Strong Entities and Weak Entities

An entity whose existence does not depend on the existence of another entity like the entity station is known as a strong entity or parent entity while an entity that depends on the existence of another entity is called a weak entity or child entity. An example of a weak entity is the entity observation. A strong entity must have a Primary Key but it is not mandatory for a weak entity to have a Primary Key.

Entity Relationships

In designing databases there is nearly always a need to define relationships between entities. For example in meteorology we know that an observation belongs to a particular station and that for one station we can have many observations, giving a one to many relationship. To design relationships in MS Access, we click of the menu item Database Tools then click on the Relationships icon on the tool bar below the menu as shown in Fig 12.

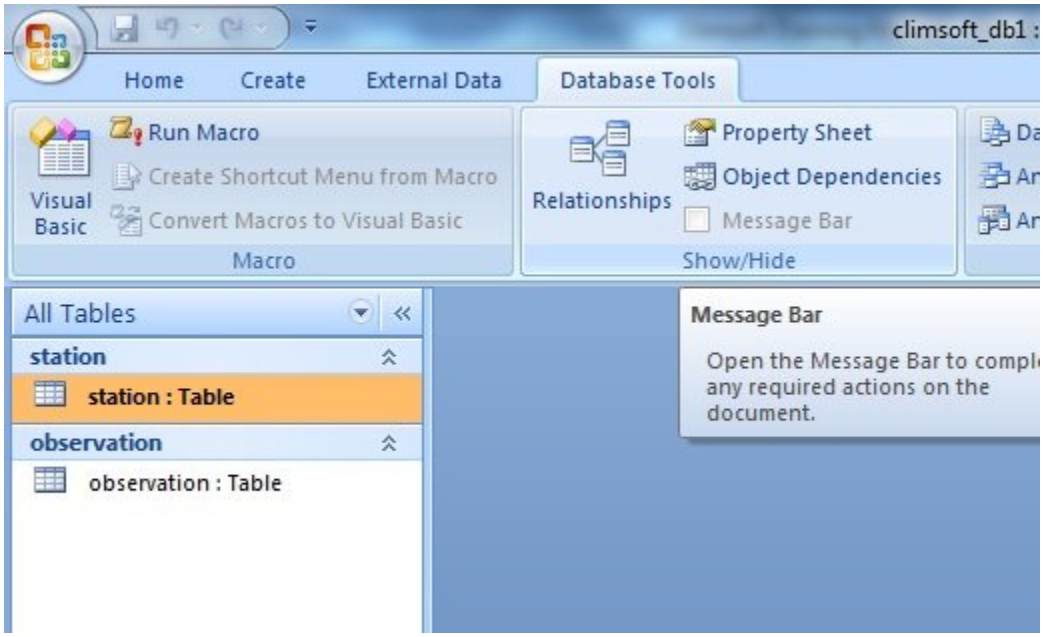


Fig 12. Tools for Designing Relationships

After clicking on the Relationships icon, a dialogue appears showing the tables available for selection in the establishment of relationships as shown in Fig 13.

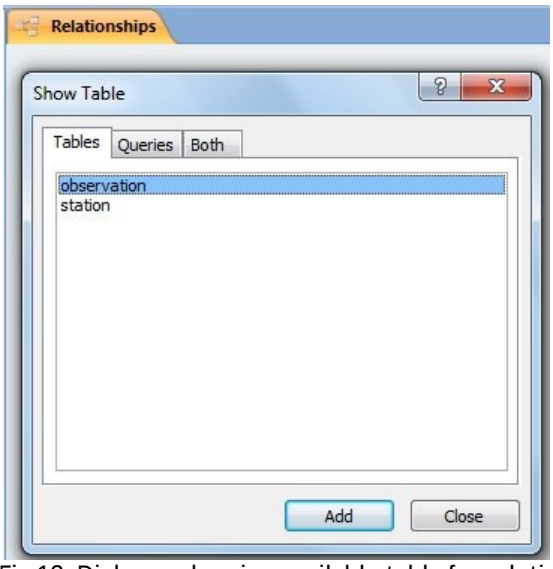


Fig 13. Dialogue showing available table for relationships

After selecting and adding the tables on which we want to create relationships, we get the dialogue shown in Fig 14.

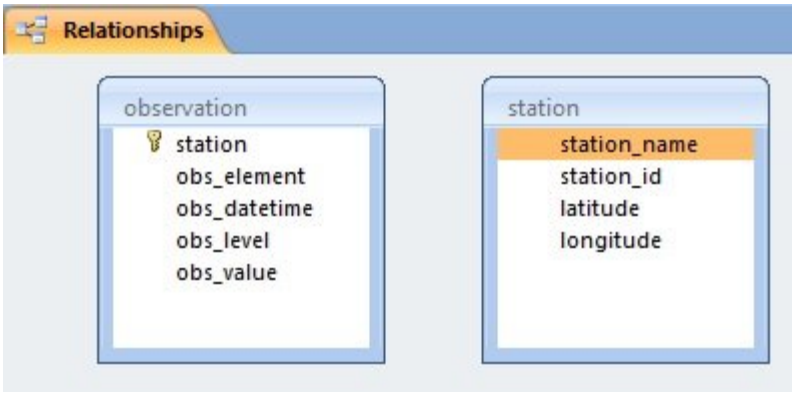


Fig 14. Tables selected for relationships

As an example, to create a relationship between the station and observation entities, we click on the Primary Key in the station table and drag this to the corresponding field (station_id) in the observation table. On releasing the mouse button, the dialogue show in Fig 15 appears.

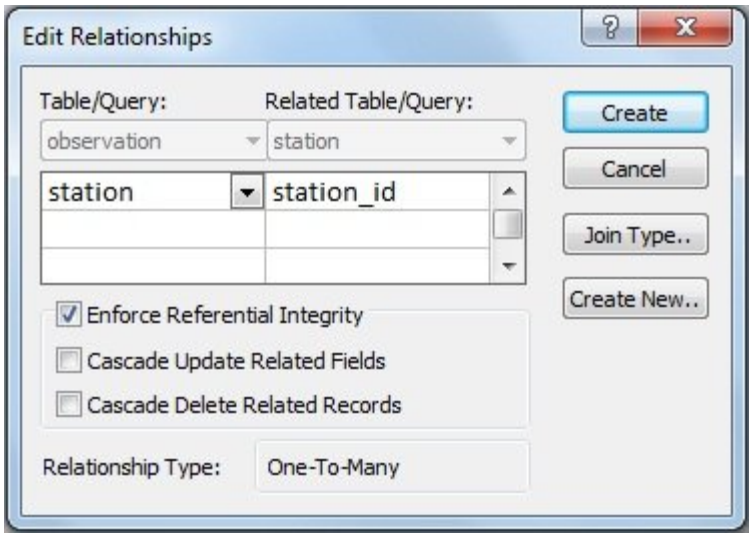


Fig 15. Creating the relationship

On selecting referential integrity and clicking the Create button, the relationship is created as shown in Fig 16. The type of relationship created in this case is a one to many relationship. This means that for one station, we can have many observations. And referential integrity means that before we can have an observation, a station must exist i.e. an observation must refer to an existing station. We cannot have an observation which is not associated with a station. The non Primary Key field *station* in the observation table that is matched to the the Primary Key field *station_id* in the station table is known as a Foreign Key.

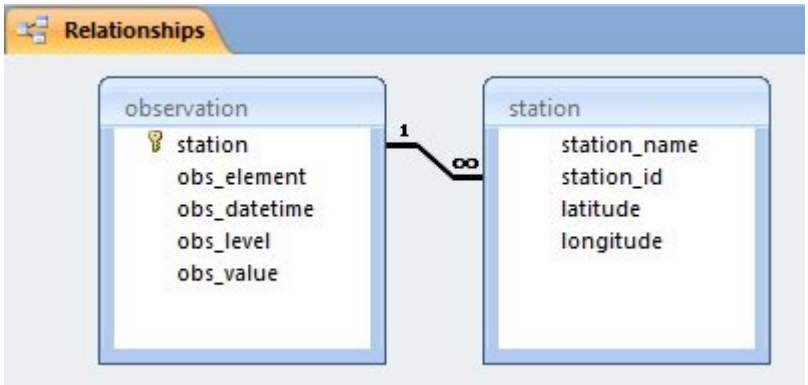


Fig 16. Relationship created

Note: Before creating a relationship graphically in MS Access, the tables must be empty. A relationship enforces a rule between the values contained in the related tables. This means that we must first enforce the rules on the data before we add the data.

Importing External Data into MS Access

Exercise 4.

To import data from a file into MS Access, you click on File → Get External data →Import. This should give you a dialogue to browse for the file you want to import. In the dialogue that follows, choose files of type Text and locate the file station_information.csv in the folder \climsoft_training\data.

You should get a new dialogue similar to Fig 17, showing a snapshot of the data contained in the file to be imported.

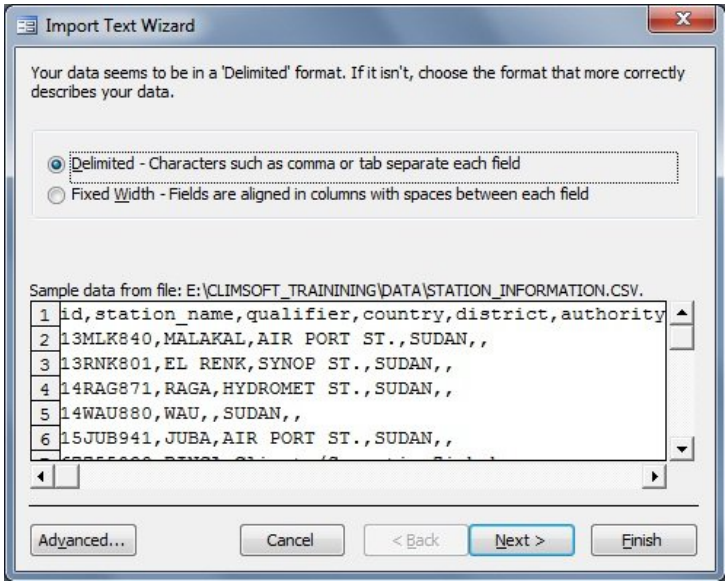


Fig 17. Snapshot of data in external csv file

After clicking Next, a dialogue similar to Fig 18 appears.

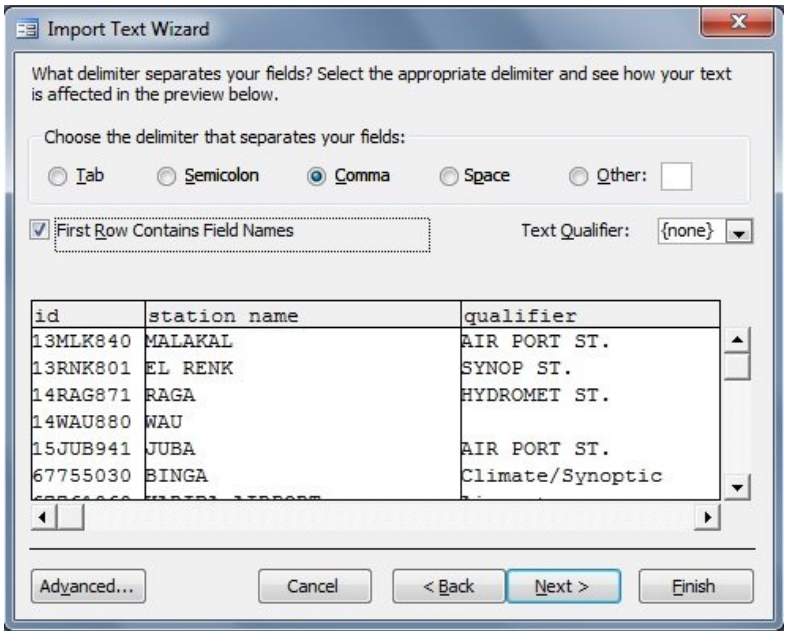


Fig 18. Specifying parameters for importing external data

Select First row contains field headings and click next. Once again click Next in the dialogue that follows. Another dialogue will appear. Select No primary key and click Next then Finish.

You should then have a new station named station_information.

Queries in MS Access

In its simplest form, a query is an object used for selecting data from a table. However, in many cases one would want to carry out other operations on the selected data e.g. adding the selected data to an existing table or modifying the selected data.

Exercise 5.

To create a new query from the MS Access main window, click on Queries under Objects and then double click on Create query in design view to get the screen shown in Fig 19.

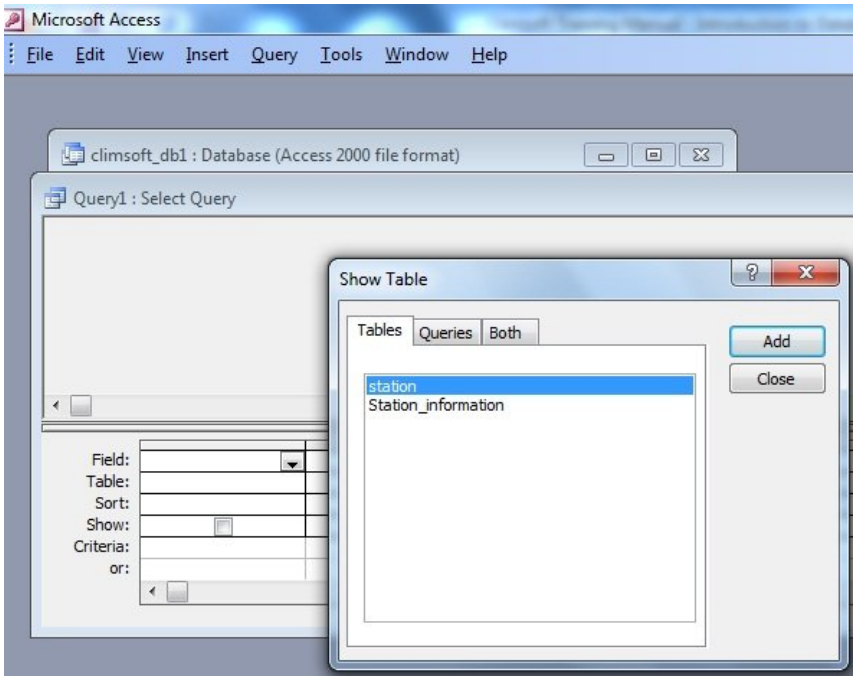


Fig 19. Designing a select query

Select the table from which you want to select data. In this case let's select the table station_information and then click Add and then click Close. This should give the dialogue show in Fig 20. There are two sections to the dialogue. The top part of the dialogue shows the table from which we want to select data and the bottom part, with rows and columns is the area where we place the fields we have selected from the table.

Double click on the field id from the table in the top part of the design window. Next, double click on the field station_name. The design window should then look like the screenshot in Fig 14. Instead of double clicking, you can also select a field by clicking on it on the top part of the design window and then drag the field to the bottom portion of the window.

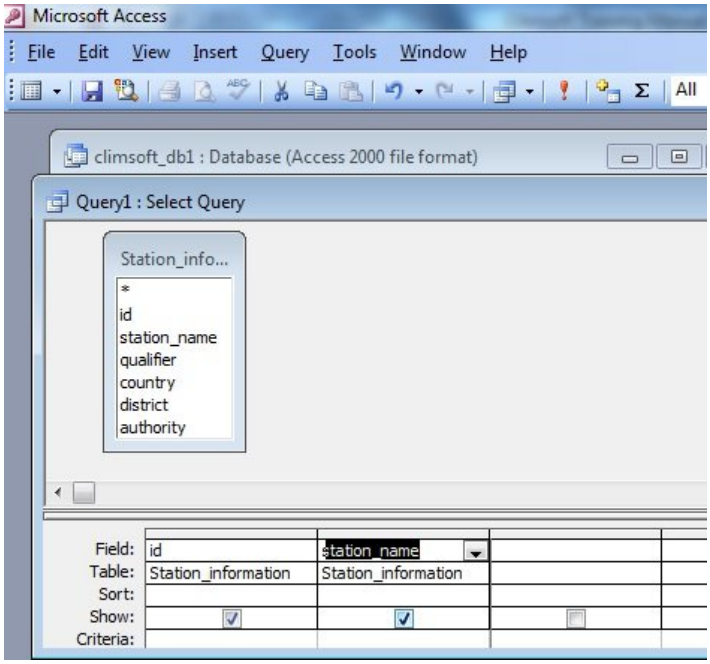


Fig 20. Selecting table fields for display

If you click on the View icon on the tool bar just below the menu bar, the selected data will be displayed as shown in Fig 15.

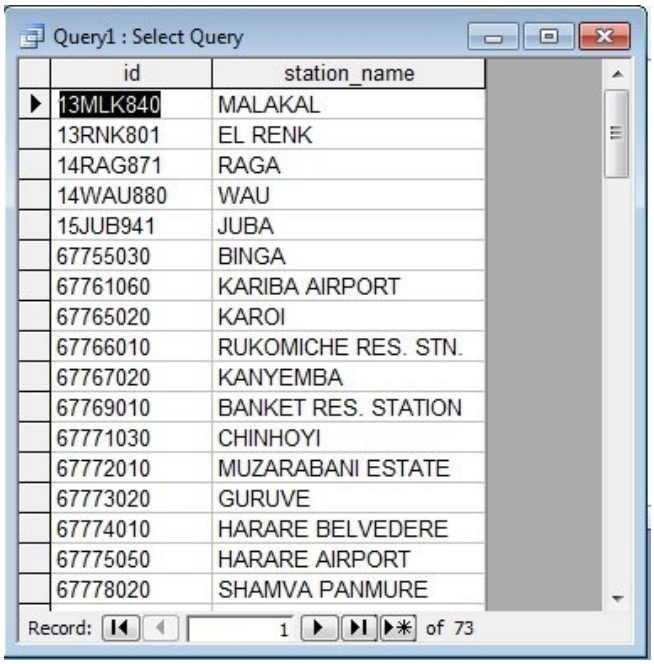


Fig 15. Display of selected data

Save the query as qry_select_stations

Exercise 6.

1. Select stations from Sudan
2. Select stations from Zimbabwe and display the fields *id, station_name, qualifier and country.*

Update Query

If we need to modify values in a table we use an update query. To create an update query, we start by following the steps for designing a select query and select the field(s) we want to modify and specify the criteria for the records to be modified, then choose Update from the Tool Bar as shown in Fig 16. The new value to which selected data should be updated to must be specified in the Update To field on the bottom part of the design window.

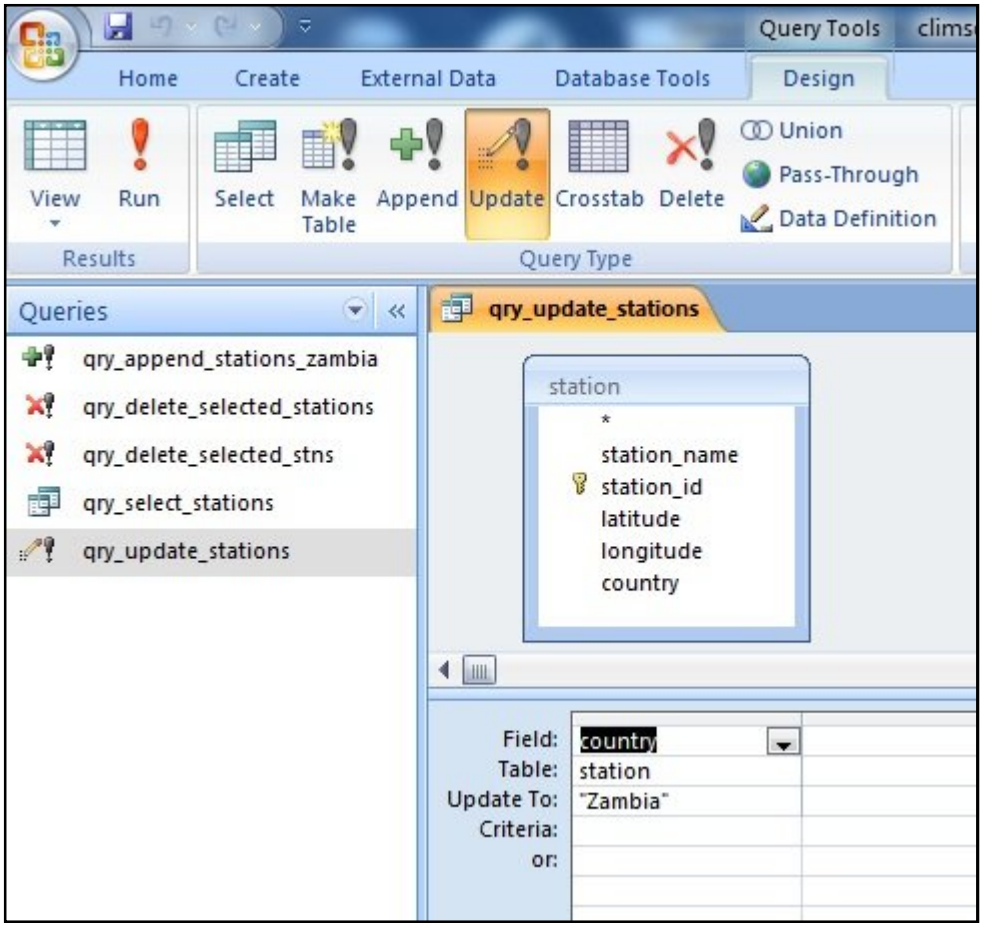


Fig 16. Designing an Update Query

Append Query

An append query is used to append data from one table to another. To design an append query, we should first design a select query to select the fields and records to be appended from the source table, and then click on Append from the Tool Bar shown in Fig 16.

After clicking on Append, a dialogue for specifying the target table will appear See Fig 17.

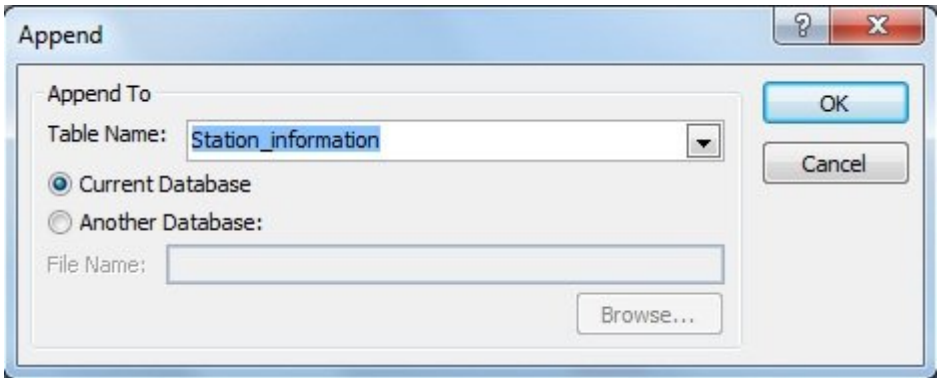


Fig 17. Specifying target table to which data should be appended

After clicking OK on this dialogue, the design window will change to give provision for specifying the target field(s) to which data from fields in the source table should be appended as shown in Fig 18.

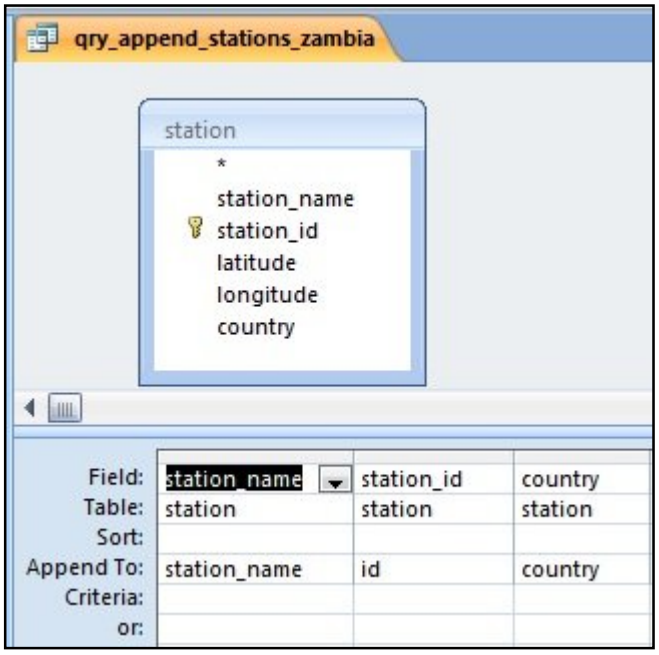


Fig 18. Specifying fields to which data should be appended

Delete Query

To design a delete query, we first design a select query to select the records to be deleted. The criteria for the records to be deleted must be clearly specified and then click on Delete from the Tool Bar. Part of the design window for a delete query is shown in Fig 19.

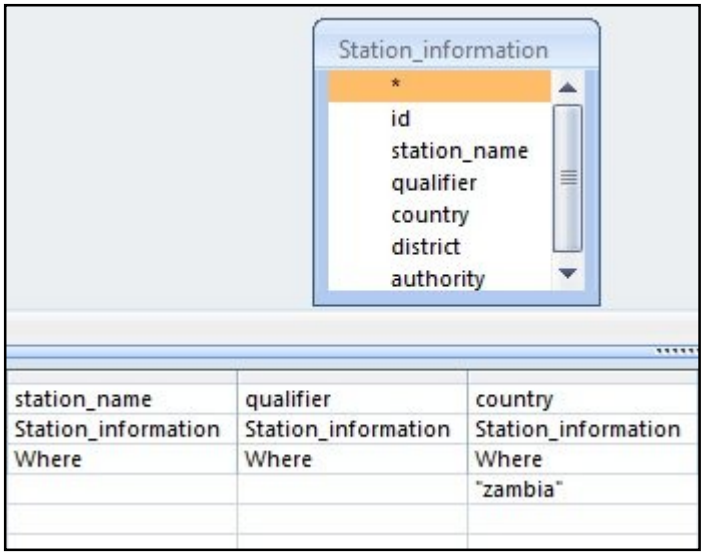


Fig 19. Delete Query criteria

Warning on Action Queries

Action queries are those queries that will make changes to data in a table e.g. Append Query, Update Query and Delete Query. You must always first view the output of an action query before executing it because the changes made to the data in the table cannot be reversed.