

Recorder 6

MS Access and *Recorder 6*

Sally Rankin JNCC Recorder Approved Expert
Written for v6.13.3 under contract to JNCC in March 2009

Contents

1	Introduction.....	2
2	Creating a version of NBNDData.mdb in a later version of Access.....	3
3	Example queries	4
3.1	Handling dates.....	4
3.2	Query to list all taxon occurrences with their location data.....	5
3.3	Query to list the number of species observations per survey.....	6
3.4	Query to list the number of observations per taxon.....	8
3.5	Query to list the number of observations per taxon plus list.....	9
3.6	Query to list the number of observations per taxon s plus group.....	10
3.7	Query to list the taxa in a rucksack.....	11
4	Conclusion	12

1 Introduction

This guide is suitable for all skill levels.

Data entered into *Recorder 6* is stored in a SQL Server database usually called NBNDData_Data.MDF. In a standard standalone installation using an MSDE instance called RECORDER this will typically be found in:

C:\Program Files\Microsoft SQL Server\MSSQL\$RECORDER\Data
along with its associated log file NBNDData_log.LDF.

Data in its predecessor, *Recorder 2002*, is stored in a Microsoft Access 97 database. Users of this system are able to use Access to query the database so in order to provide the same functionality in *Recorder 6* the installation process creates an Access 97 database called NBNDData.mdb or nbndata.mdb which is stored in the *Recorder 6* Database folder, typically:

C:\Program Files\Recorder 6\Database in a standalone installation.

It contains links to the tables in the *Recorder 6* SQL Server database and can be used in much the same way as using Access on the *Recorder 2002* database. The SQL Server database itself can also be viewed and queried using tools such as Enterprise Manager (SQL Server 2000) or SQL Server Management Studio (SQL Server 2005) or SQL Server Management Studio Express.

Reporting in *Recorder 6* will meet many of a user's needs, but perhaps not all, so being able to supplement the reporting capabilities via Microsoft Access may prove useful. However, the database structure is very complex and will require time and effort to master. It contains over 150 related tables. The design is based on the NBN Data Model, documentation for which is available in the Documentation Wiki on the *Recorder 6* web-site www.recordersoftware.org. The links to follow are Main Page – Data Models (not Recorder 6 – Data Model). As with *Recorder 2002*, care must be exercised when using the database outside *Recorder 6* as users could easily alter the data in a way that will cause problems with running the system. Some such problems may prove very serious, so don't modify anything unless you are sure you know what you are doing and unless you are prepared to test the changes very thoroughly to ensure they don't cause problems with any other aspect of the system. If you are concerned about this you are advised to restrict the use of the Access database to reporting.

In *Recorder 2002*, the dictionaries (taxon, biotope and admin areas) and the associated index files are held in separate Access 97 databases, nbndict.mdb, Index_Taxon_Group, Name & Synonym.mdb, whereas in *Recorder 6* these are in the same database as the observations and locations, etc.

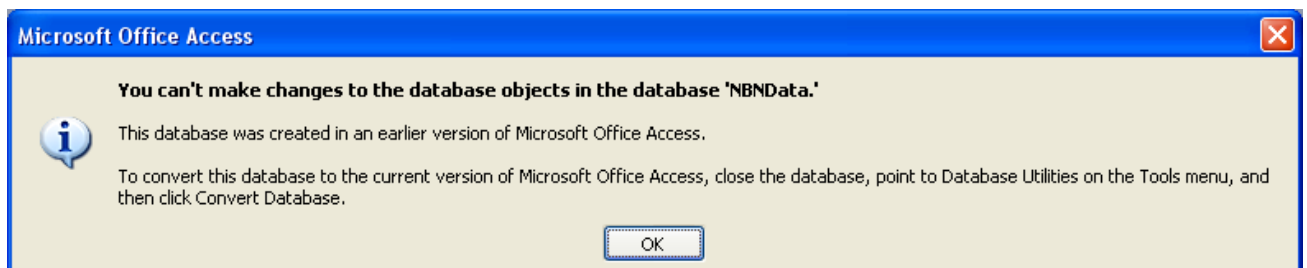
Please note that it is not uncommon for a *Recorder 6* installation to fail to create a working version of *NBNDData.mdb*. A tool is available for creating it in these circumstances – please contact the supplier of your copy of *Recorder*, or any of the *Recorder* suppliers listed on the *Recorder* Software web site www.recordersoftware.org for details. Also, the Network Installation Guide recommends that it be removed if users are concerned about the security threat it poses.

Users can also create their own ODBC connection to the *Recorder 6* SQL Server database. For an example, see the article called ‘Using the Reporting Snapshot Tool to link Recorder 6 to MapInfo’ in the Documentation Wiki on the *Recorder 6* web-site www.recordersoftware.org. The links to follow are: Documentation Wiki – Recorder 6 – Third-party Tools.

This document is designed to be a basic introduction to using Access with *Recorder 6* for users familiar with Access

2 Creating a version of *NBNDData.mdb* in a later version of Access

If you open an Access 97 database with a later version of Access, you will get a message like:



This means that you can look at the data in any of the objects, e.g. tables and queries that already exist, but you can't design new queries etc. To design your own queries you need to create a version of this database in the version of Access that is on your system:

- Open Microsoft Access
- Select **Tools – Database Utilities – Convert Database – To Access 2002 – 2003 File Format...** (for example)
- Navigate to **NBNDData.mdb** in *C:\Program Files\Recorder 6\Database*, or the equivalent on your system, and click **Convert**
- Navigate to where you want to store the new database, enter a name for it, say **NBNDData A2003.mdb** and click **Save** and **OK**

You will now have an Access 2003 version of *NBNDData.mdb* in which you can design queries to report on the database. If you retain *NBNDData.mdb* you will have a version you can return to in case of problems.

N.B. The instructions given above are for Access 2003 – they may vary in other versions of Access.

In *Recorder 6*, there is a mechanism for creating XML reports which uses SQL statements within a standard XML structure. This method allows users to create reports and share them with others. Access queries can be used to generate the SQL in XML reports. This is very helpful for users who aren't very familiar with SQL, but once people get some experience they often prefer to write the SQL themselves rather than use Access. A variation on XML reports called XML batch updates can

be used to make changes to the database. A number of XML reports and batch updates are made available with *Recorder 6* when it is installed.

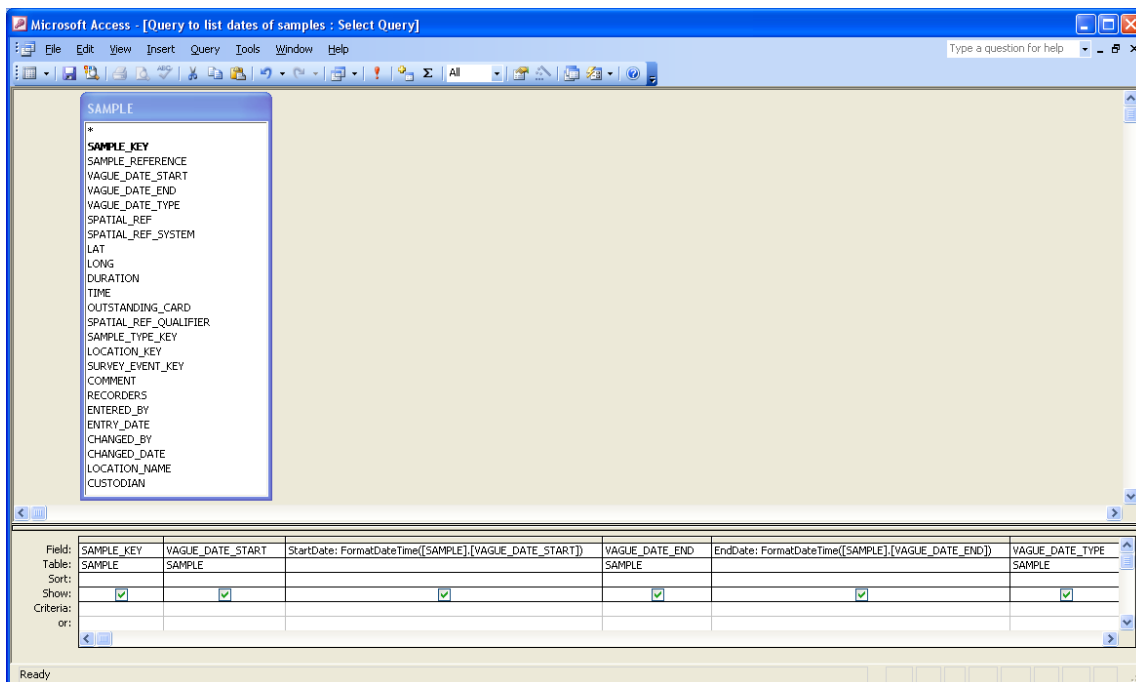
For a brief introduction to writing XML reports see Help – Tasks on the Contents tab – Reporting – XML reports. There is, however, much more information on the Documentation Wiki, and Mike Weideli, another JNCC *Recorder* Approved Expert, has made more available via the link in his post on the forum on 6/07/08 – <http://forums.nbn.org.uk/viewtopic.php?id=686#p3058>.

3 Example queries

3.1 Handling dates

Dates for observations in *Recorder* can be input in a variety of ways which collectively are called ‘vague dates’. Internally, these are held in three fields: *Vague_date_start*, *Vague_date_end* and *Vague_date_type*. The start and end dates represent the first and last day of the date range for the vague date, and the type field is an indicator as to what kind of date it is (a day, day range, a month, season, year, etc.). For more information, see the Wiki on the *Recorder 6* web-site www.recordersoftware.org. The links to follow are: Documentation Wiki – Recorder 6 – Data Entry – Vague Dates.

Recorder 6 stores the start and end dates of vague dates as integers which will need to be converted to dates in the format dd/mm/yyyy for most purposes. This can be done using the *FormatDateTime* function as illustrated in the query below. The functions illustrated on the Wiki are for use in SQL Server, not Access.



Screen shots like this will be easier to read if you set the zoom to 150%.

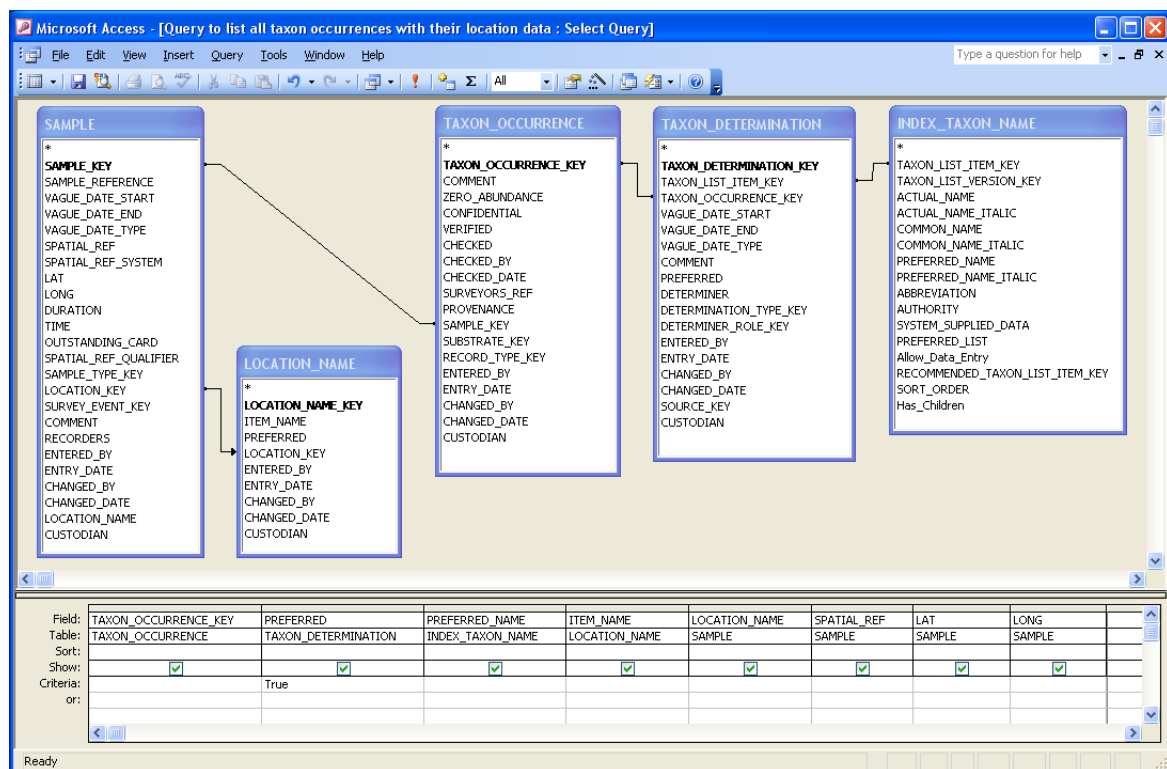
The format of the function is `StartDate: FormatDateTime([SAMPLE].[VAGUE_DATE_START])`

An extract of the result is:

Query to list dates of samples						
SAMPLE_KEY	VAGUE_DATE_START	StartDate	VAGUE_DATE_END	EndDate	VAGUE_DATE_TYPE	
SR00000100000C0F	39203	01/05/2007	39203	01/05/2007	D	
SR00000100000C0G	39223	21/05/2007	39229	27/05/2007	DD	
SR00000100000C0H	39203	01/05/2007	39233	31/05/2007	O	
SR00000100000C0I	39234	01/06/2007	39325	31/08/2007	P	
SR00000100000C0J	39448	01/01/2008	39813	31/12/2008	Y	
SR00000100000C0K	39083	01/01/2007	39813	31/12/2008	YY	
SR00000100000C0L	0	00:00:00	39447	31/12/2007	-Y	

3.2 Query to list all taxon occurrences with their location data

This query shows how to extract all records with their location data for use in a GIS system.

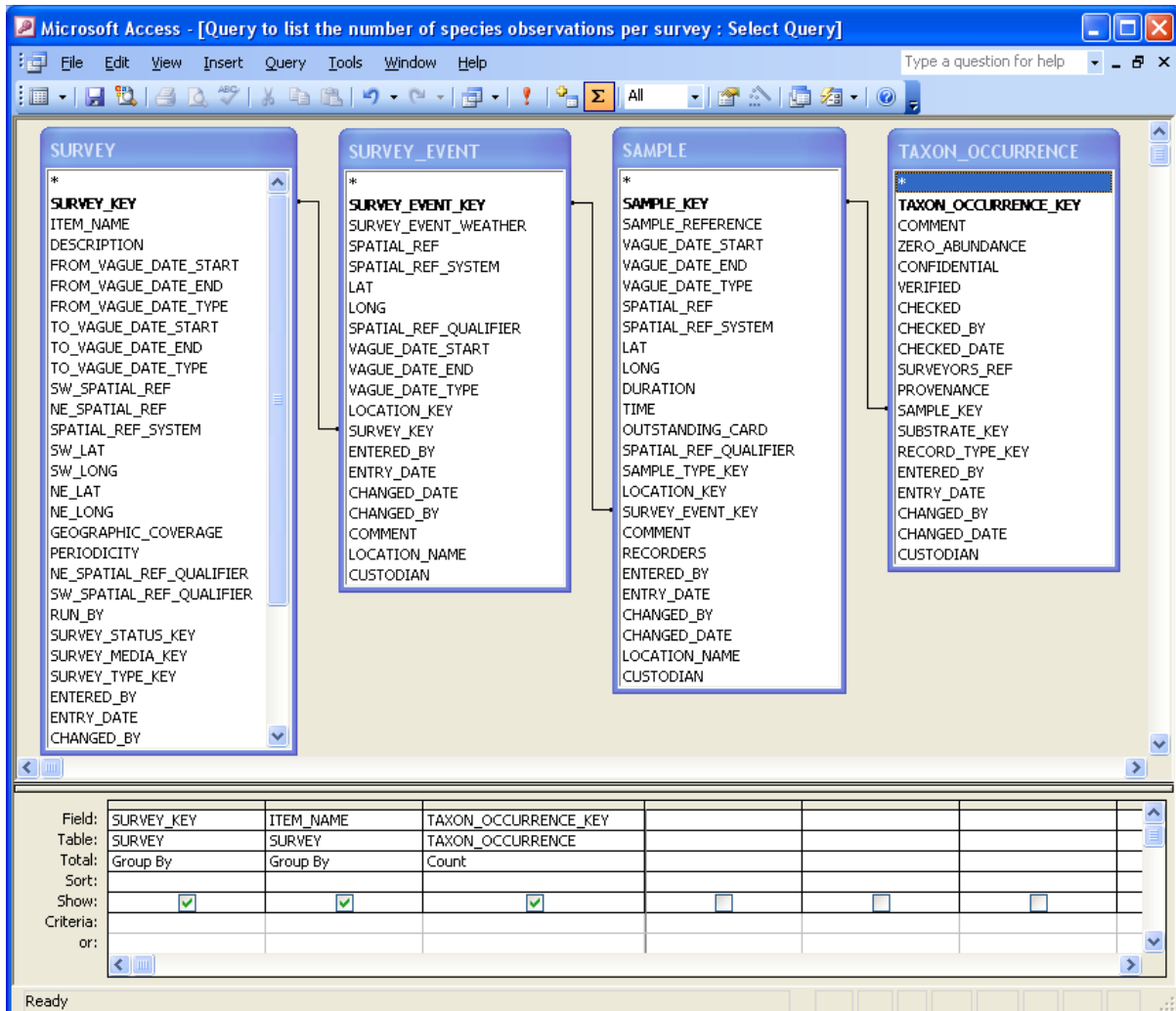


The name of the location can be obtained directly from Location_name without going via the Location table, as illustrated above, but the link between Sample and Location_name is not automatically created so you need to insert it manually. Also, it must be changed to an outer join to prevent Samples not linked to Locations from being excluded from the report.


Without the criteria of Preferred = True, the query will list all Taxon_determinations so Taxon_occurrences (species observations) with multiple determinations will be listed once for each determination. Each Taxon_occurrence will have one or more Taxon_determination of which one will be flagged as preferred, i.e. it will be ticked on the Taxon_determination tab that users can view in the observation hierarchy in Recorder 6.

If any locations have multiple names you will need to add the criteria of Preferred = True for Location_name as well.

3.3 Query to list the number of species observations per survey



Every species observation entered into *Recorder* will have a row in the *Taxon_occurrence* table, hence a count of the taxon occurrences per survey will provide the number of species observations per survey.

If you want to convert a query like this into an XML report so that you can run it in *Recorder 6* itself, you can extract the SQL by selecting **View – SQL View** or by clicking the View down arrow  and selecting SQL View, then copying the SQL displayed into the required XML structure.

In this case, the SQL is:

```
SELECT SURVEY.SURVEY_KEY, SURVEY.ITEM_NAME,
Count(TAXON_OCCURRENCE.TAXON_OCCURRENCE_KEY) AS
CountOfTAXON_OCCURRENCE_KEY
FROM ((SURVEY INNER JOIN SURVEY_EVENT ON SURVEY.SURVEY_KEY =
SURVEY_EVENT.SURVEY_KEY) INNER JOIN SAMPLE ON
SURVEY_EVENT.SURVEY_EVENT_KEY = SAMPLE.SURVEY_EVENT_KEY) INNER JOIN
```

```
TAXON_OCCURRENCE ON SAMPLE.SAMPLE_KEY =  
TAXON_OCCURRENCE.SAMPLE_KEY  
GROUP BY SURVEY.SURVEY_KEY, SURVEY.ITEM_NAME;
```

When copied into the required XML structure and reformatted to make it a little easier to read the XML report will be:

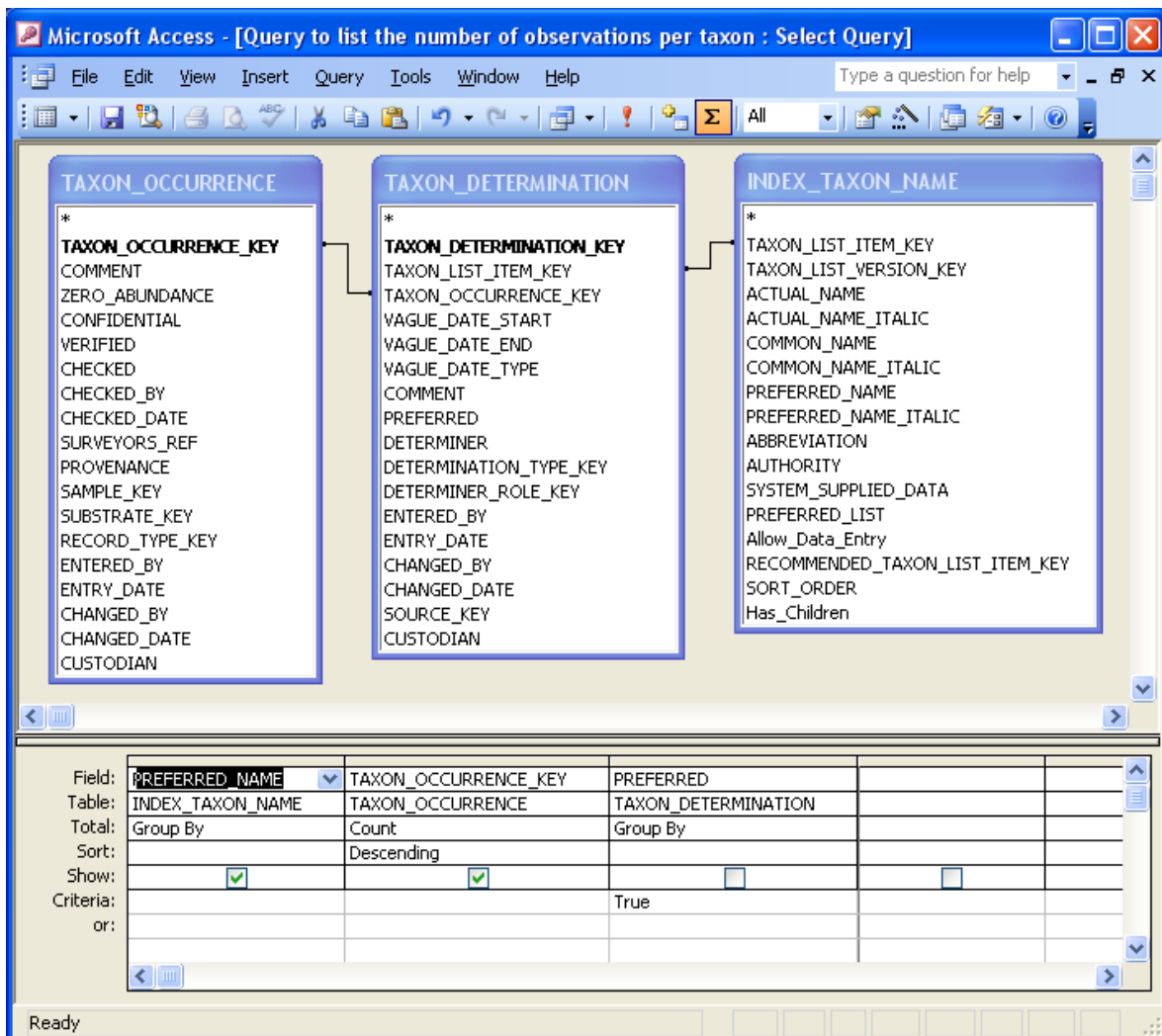
```
<?xml version="1.0" ?>  
  
<CustomReport menupath="Statistics" title="No of species observations per survey"  
description="Query to list the number of species observations per survey">  
  
<SQL>  
  
SELECT  
  
SURVEY.SURVEY_KEY,  
SURVEY.ITEM_NAME,  
Count(TAXON_OCCURRENCE.TAXON_OCCURRENCE_KEY) AS  
CountOfTAXON_OCCURRENCE_KEY  
  
FROM ((SURVEY INNER JOIN SURVEY_EVENT ON SURVEY.SURVEY_KEY =  
SURVEY_EVENT.SURVEY_KEY)  
INNER JOIN SAMPLE ON SURVEY_EVENT.SURVEY_EVENT_KEY =  
SAMPLE.SURVEY_EVENT_KEY)  
INNER JOIN TAXON_OCCURRENCE ON SAMPLE.SAMPLE_KEY =  
TAXON_OCCURRENCE.SAMPLE_KEY  
  
GROUP BY SURVEY.SURVEY_KEY, SURVEY.ITEM_NAME;  
  
<Where Keytype="Default">  
  
</Where>  
  
</SQL>  
  
</CustomReport>
```

If this is saved as an XML report as described in the *Recorder 6* Help – see Help – Tasks on the Contents tab – Reporting – XML reports, you will be able to run it from within *Recorder 6*.

Note:

- Including an ORDER BY clause will enable you to order the result by survey name or the count of taxon occurrences instead of the survey key
- Including Columns clauses will enable you to change the column headings and the width of the columns
- This is a very simple example designed to illustrate the process and help users get started with XML reports. You will often find that the SQL will need modifying before it will work in an XML report
- Users with some Access expertise will be able to produce simple XML reports like this but for more complex reports you may need assistance – please contact the supplier of your copy of *Recorder*, or any of the *Recorder* suppliers listed on the *Recorder* Software web site www.recordersoftware.org if you would like help.

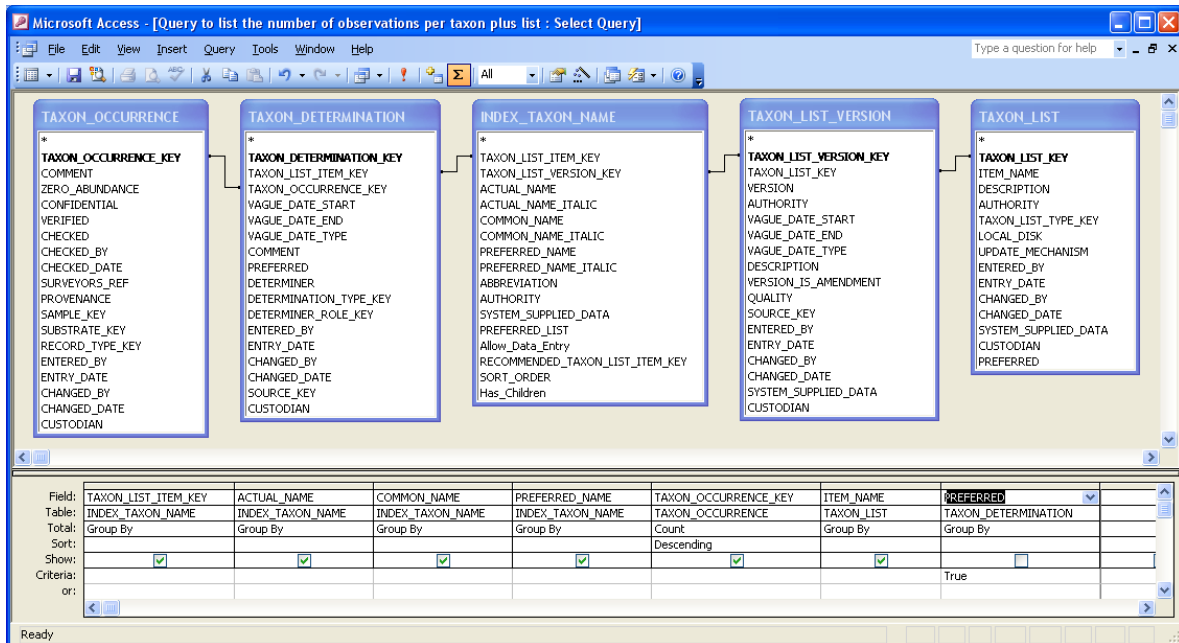
3.4 Query to list the number of observations per taxon



This query lists all taxa in the database for which there are observations, and the number of observations (taxon occurrences) for each.

When designing this query the link between Taxon_determination and Index_taxon_name is not automatically created so you need to insert it manually.

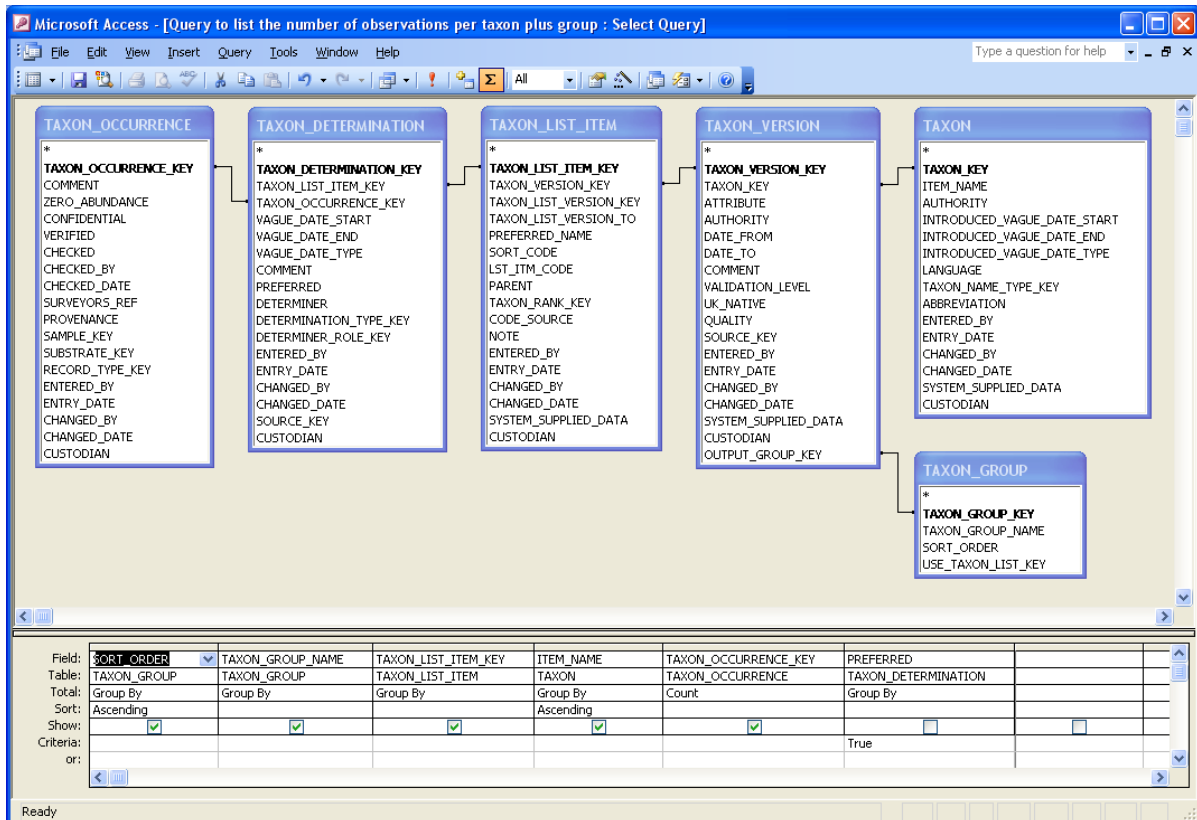
3.5 Query to list the number of observations per taxon plus list



As in the previous query, this one lists all taxa in the database for which there are observations, and the number of observations (Taxon_occurrences) for each. However, it includes the taxon list so each taxon will be listed once for each list in the taxon dictionary that has been used to enter observations for it. The Taxon_list_item_key is the unique identifier for a particular species on a particular list (Item_name) in the taxon dictionary. It is the entry that will appear in Rucksacks and Recording Cards in Recorder 6 if the associated species is added.

Index_taxon_name is an index on the taxon dictionary designed to speed up use of the dictionary. It contains the species names although it contains three names for each key: Actual_name, Common_name and Preferred_name. The Actual_name and Preferred_name can be used to identify synonyms: if the Actual_name is a scientific name and is not equal to the Preferred_name, this means that the Actual_name is a synonym of the Preferred_name. If the Actual_name is equal to the Common_name, observations entered using the corresponding Taxon_list_item_key will have been entered using the common name instead of the scientific name.

3.6 Query to list the number of observations per taxon s plus group



This query also lists all taxa in the database for which there are observations, and the number of observations (Taxon_occurrences) for each. However, it includes the taxon group (Output_group_key) from the Taxon_version table so it uses the taxon tables from the taxon dictionary (Taxon_list_item, Taxon_version and Taxon) instead of the index (Index_taxon_name).

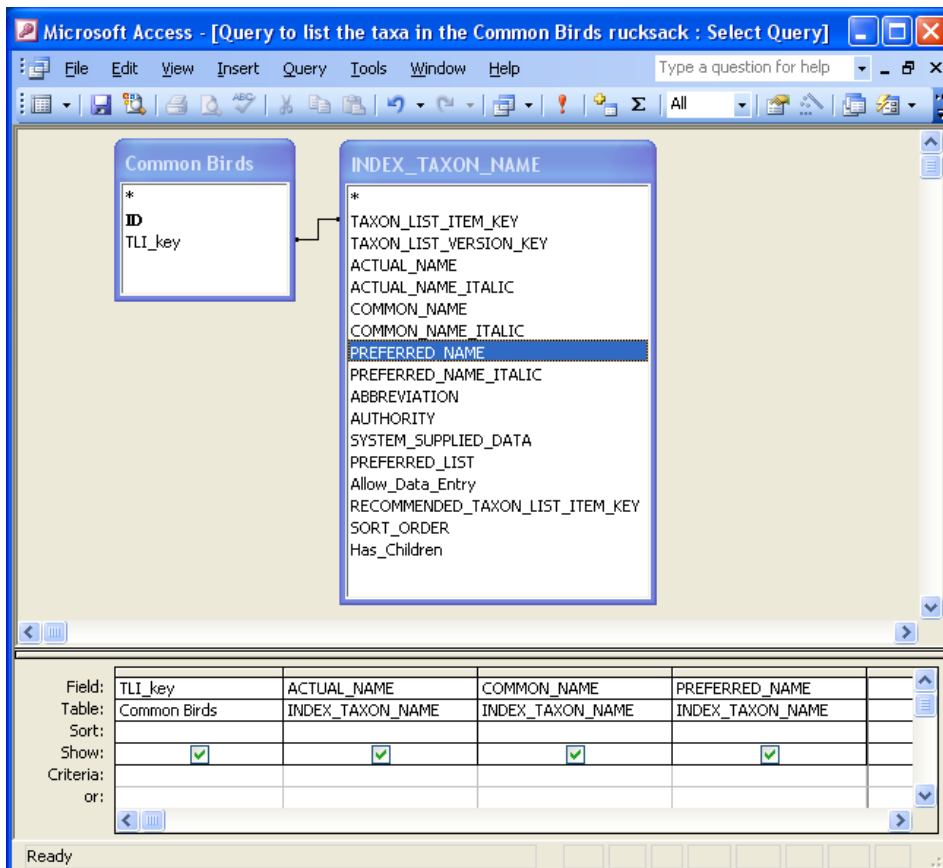
Note that if users add taxa to the dictionary using **Dictionaries – Edit Taxon Details** there is no field for entering the taxon group (the Taxon Group tab is a different facility) so the Output_group_key on Taxon_version is left blank. Hence in the query, the join between Taxon_version and Taxon_group must be changed to an outer join if your system contains user added taxa, otherwise they will be excluded from the report. Alternatively, you could look up the taxon group in the Taxon_group table and add its key to the Taxon_version entry for the user added taxa concerned. It is to be hoped that this bug will be corrected in a future release.

When designing queries like this it is a good idea to check that it is returning the correct information. One way of doing this is to copy the result into Excel and total the count column. This total should be equal to the number of rows in the Taxon_occurrence table. If it isn't, it probably means there are taxa present with Output_group_key blank, which need handling as specified in the paragraph above.

3.7 Query to list the taxa in a rucksack

It isn't possible to print a rucksack containing taxa in *Recorder 6* itself or to list the taxa in, say, Excel. Also, if you open the file in a text editor like Notepad or Microsoft Word you will see that it only contains the internal codes for the taxa. not their names. These codes are Taxon_list_item_keys. However listing the names can be done in Access as follows:

- Take a copy of the rucksack and change its file extension from .Ruk to .txt
- Import the required rucksack into your Access database by selecting the Tables tab then **File – Get External Data – Import**.
- Navigate to where the rucksack is stored, select Text Files for Files of Type, select the .txt version of your rucksack and click Import.
- Click Next at each stage of the Import Text Wizard, then Finish at the end. A message should then appear saying that the Wizard has finished importing the file.
- Now design the following query:



The tags in the rucksack, e.g. <TAXON> and </TAXON>, will effectively be ignored as there won't be any matches for them in Index_taxon_name, but any codes between the other tags will need to be removed if they return incorrect matches.

If criteria of [INDEX_TAXON_NAME]![ACTUAL_NAME] = [INDEX_TAXON_NAME]![PREFERRED_NAME] is added to this query you can test whether the rucksack contains any synonyms. If the number of rows that result from running it is the same with or without this criteria then there are no synonyms in the rucksack.

4 Conclusion

With a little effort devoted to understanding the *Recorder* data model, users with some Access expertise will be able to query the database using NBNData.mdb, the Access 97 database supplied when installing *Recorder 6*. It isn't necessary to understand all aspects of the model prior to starting to query it as any query will usually only use a smallish number of tables. Most tables are sensibly named so if you know what things are called in the front end of *Recorder 6* it should be relatively easy to find the tables that contain that data in the database.

The queries in this document are designed to provide a few introductory examples to help get users started with querying the *Recorder 6* database. If you need assistance with more complex queries, please contact the supplier of your copy of *Recorder*, or any of the *Recorder* suppliers listed on the *Recorder* Software web site www.recordersoftware.org.