DB for new data analysts + considerations on between individual consistency in data extraction

1. From: Isabel Winney (‘Quirky sparrows’ post on the sparrow google group, 23/04/2014): The database folder is named **'DataBaseV0.xx-date'**. The V0.whatever number is the number given by Ian Stevensson to the version of the code creating the data base ([ian@sunadal.co.uk](mailto:ian@sunadal.co.uk), contractor to be paid if required to change the DB code). The date is like our version number, and is the date when we last entered or changed data.

Our database is split in to three sections and one system folder. One section is **'SparrowDatabase0.xx'** which is the front end, or the human interface. This part stores queries and links together tables and is not changed when data is entered. The second section is **'SparrowData'**. This is where measurement and event data is stored and therefore this DOES change when data is entered. We keep regular back-ups of this part when we input data in the summer. The third section is **'SparrowGenetics'** and this also changes when data is entered but not so much. It is where the microsat data is stored. The system folder is essential running info, so don't touch.

1. Each table purpose is described in ‘**tblTableList’**, each field from each table is described in ‘**tblFieldList’** as well as in the ‘**Data Dictionary’** found under the maintenance tab of the sparrow menu. The values or codes that can take each field are specified in the ‘**tblCodeList’** and most are also given in the ‘**Data Dictionary’**. Links between tables are most easily seen from ‘**relationships’** found in the tab ‘Database Tools’.
2. They are other data that exists but which are **not included** in the database, or at least not in their updated version. This include the genetic **pedigree**, the raw data for **provisionning**, **personality** measurement, etc. These can be found, at the moment, in the dropbox folder.
   * *‘****tblGenPedigree’*** *does not seem to be used by the group, I suggest to update this table in the common DB when new pedigrees are made, since many ‘****sys\_queries’*** *are based on it.*
3. To get to the SQL code of ‘**sys\_queries’**, you can go to ‘file’, ‘options’, ‘current database’, ‘navigation options’, ‘display options’, tick ‘show system objects’. The system objects comprise ‘usys\_tables’ and ‘usys\_queries’, the later being used to build the ‘sys\_queries’. Those queries also use values set in ‘**tblDatabaseSettings’**.

* *These* ***sys\_queries’*** *were made by Ian, they do not seem to be used by the group.*

*What about including a description of what the query does, giving precisions on which data were or not included. This could be used for consistency between group members. For each new query designed by each member for a specific question, one could have a ‘sys\_query’ as a start, and one could justify changes such as the inclusion or rejection of specific data.*

*I feel that there should be some consensus on having for instance an unknown (NA) field value (e.g.: sex, WPO or EPO, Social Dad, etc), which could be the default in a ’sys\_query’, and that one should always be able to explain why, when adressing a specific question, one use a larger or more restricted dataset. We might be getting about the same output with each of our query to address a question (e.g.: last date seen alive), but they are clearly inconsistencies arising from subtle decisions, which can have a lot of impact.*

*Importantly, subtle decisions that need to be made when building a query might not be obvious for people who haven’t collected the data targeted (which is, at various degree, the case for all of us, as it is a long term monitoring).*

*Of course, always making your own queries would reduce the propagation of error, but one specific query might be a huge mistake and it is hard for the others to give feedbacks on that. Or is it not ?*

*Equally, one should always look at the details of a sys\_query before actually using it in a to-be-published analysis; a consensual sys\_query wouldn’t be meant to be ran carelessly, which I think is here a minor risk compare to the benefits of having between-team members consistency and an easy way to give feedbacks on each other’s queries/decisions.*

1. I provide my annotated SQL codes dissecting those sys\_queries. They can be used for learning SQL and smart ways of querying things, for understanding the subtlety of the DB, for inspiration in writing your own queries for a related question.
2. For some complex queries, I made a power point hierarchical organization chart to see the structure of one query and all the subqueries it is based on. Dark blue are for queries, dark green for tables, others colors are for redundant subqueries either only explained once within a sys\_query scheme, or explained in other sys\_query scheme.
3. For the query Sys\_SexEstimate, an extra R file was create to give an example of how to call Access from R (which means that it always gets up to date data), and insert your SQL queries in R. In addition, there is a small tutorial R, how to write SQL in R.

* *FYI, I myself read .SQL and .R files with NotePad++. Annotations in SQL language are preceded by --, annotations in R are preceded by #. Let me know if this isn’t compatible with R studio or anything you might be using.*