obslog - an elementary database model for logging astronomical observations

Hernán De Angelis*

2016-12-16

Contents

*Epistulae Astronomicae

1 About *obslog*

What happens when you have accumulated over 20 years of observations as an amateur astronomer? If you have a sort of logbook you will avoid forgetting what you saw, where, when and with whom. However, a physical record can become hard to search and classify as it grows large. It is for this reason that I created *obslog*, a relational database for storing astronomical observations of the kind made by amateur astronomers. It was born out of the need for the long-term preservation and retrieval of my records as an amateur astronomer, for planning future sessions, and for gladdening my heart on some future day when the amount of astronomical observations and sessions will be (hopefully!) so vast as to overwhelm my poor old memory.

2 Concept

The central idea in *obslog* is that everything an amateur astronomer does, like visual observations, photography, variable star estimations, spec-

troscopy, etc., takes place during a *session*. A session occurs during a specific span of time, in a specific *site*, under a given weather, Moon phase and particularities. These properties are common to all activities that are carried on during a session and do not need to be entered more than once, that is, as a row in a table describing sessions.

In the same way, activities that amateur astronomers do during a session have different sets of properties that define them, although some elements might be common. For example, all observations have a target *object*, are done with an *instrument*, which can be fitted with an *eyepiece*, *filter*, *accessories*, etc., and result in some kind of *note*, *sketch* or data.

In principle, all this might be easily stored in spreadsheets, which is in fact what many amateurs do. I nevertheless chose a relational database model because it is more flexible and powerful approach than a spreadsheet while being only a bit more complicated. In a relational database entities are described non–redundantly in tables and the retrieval of information is done through queries. Queries can be arbitrarily complicated and may invole information stored in many tables, a procedure called table joins.

Following the relational database paradigm, in *obslog* redundancy is minimized by defining separate tables for specific classes of activity and equipment, like session, site, observation, object, instrument, eyepiece, etc. Data are stored once, in only one place, and given an unique identifier. Every specific item can thus be uniquely referred to in another table if necessary, minimizing the chance of errors, and when something needs to be changed it only needs to be changed once at the place where it is stored. As it is not always possible or desirable to fill all fields, I deliberately chose to avoid the `not null` clause in most fields except for the unique identifier (`primary key`). Also following the relational database model, anything that can be calculated from stored data is explicitly avoided, for example the magnification given by a certain combination of telescope and eyepiece, and the Julian date, which if needed can be calculated in suitable queries.

The current design of the database necessarily reflects my style of practicing the hobby: mostly observational sessions, including visual variable star estimations, with occasional imaging and sketching. I therefore included some tables with information from the American Association of Variable Star Observers – AAVSO that are only useful for variable star observers. Obviously, this layout might not be adequate for whom imaging is the en-

tire purpose of her/his sessions. This should not be a problem, as *obslog* is free software and users have the code at their disposal to make modifications that satisfy their needs.

3 System considerations

My implementation of *obslog* is done in SQLite. I chose this package because of its simplicity and portability, as the complete database fits within a single file. The code to generate the tables is nevertheless written in standard SQL and can be used in any other relational database system. The sample file was generated in Linux (openSUSE Leap 42.1) but the SQL code should work in any other operative system.

4 Tables

4.1 Main tables

The main tables in *obslog* are the `session` and `observation` logs. In addition, I added two other main tables to suit my needs: the variable star observation log (`varstarlog`) and the imaging log (`imglog`). These tables can only work properly with the information stored in the auxiliary tables, described in the next subsection. Data types for each column in the tables is given in parentheses after the description.

4.1.1 session: session table

- sessionID (integer) unique identifier for the session, autoincrement (`PRIMARY KEY`).
- sessionSite (text) unique identifier for the site as defined in the site table (`FOREIGN KEY`).
- sessionDateTimeStart (text) a string with ISO 8601 format, e.g. `2016-07-30 22:55`. This is data type text because date is not defined in SQLite. I always use local time.
- sessionDateTimeEnd same as above.
- SummerTime (boolean – Y/N) if summer time was valid in the site at the time of observation.

- sessionTempC (real) air temperature in degrees Celsius.
- sessionRHproc (real) air relative humidity.
- sessionWindSpeedMS (real) wind speed, in meters per second.
- sessionWindDirAz (real) wind direction, azimuth.
- sessionCloudFraction (real) fraction of the sky covered by clouds.
- sessionCloudType (text) cloud type.
- sessionSeeingAnt (integer) approximate seeing, Antoniadi scale.
- sessionTransparency (integer) scale from 1 to 5, 1: poor, 2: acceptable, 3: good, 4: very good, 5: excellent.
- sessionMoonPhase (real) Moon phase as fraction, from ephemeris.
- sessionNotes (text) any other important notes for the session.

4.1.2 obslog: observation log table

- obsID (integer) unique identifier for the observation, autoincrement (`PRIMARY KEY`).
- obsSession (integer) session ID, unique identifier of session in which the observation takes place as indicated in the auxiliary session table (`FOREIGN KEY`).
- obsDateTimeLocal (text) a string with ISO 8601 format, e.g. `2016-07-30 22:55` .
- obsObject (text) target, unique identifier for object as indicated in the astronomical object table (`FOREIGN KEY`).
- obsNotes (text) observation notes, free text.
- obsSketch (text) in my case usually a pointer to physical book and page where the sketch can be found.
- obsInstrument (text) unique identifier to instrument, as indicated in the auxiliary instrument table (`FOREIGN KEY`).

- obsMount (text) unique identifier to mount, as indicated in the auxiliary mount table (`FOREIGN KEY`).
- obsEyepiece (text) unique identifier to eyepiece, as indicated in the auxiliary eyepiece table (`FOREIGN KEY`).
- obsLens (text) unique identifier to lens (e.g. Barlow), as indicated in the auxiliary lens table (`FOREIGN KEY`).
- obsFilter (text) unique identifier to filter, as indicated in the auxiliary filter table (`FOREIGN KEY`).
- obsAccessory (text) unique identifier to accessory, as indicated in the auxiliary accessory table (`FOREIGN KEY`).

4.1.3 imglog: imaging table

- imgID (integer) unique identifier for the imaging session, autoincrement (`PRIMARY KEY`).
- imgSession (integer) session ID, unique identifier of session in which the imaging takes place as indicated in the auxiliary session table (`FOREIGN KEY`).
- imgDateTimeStart (text) start time, a string with ISO 8601 format, e.g. `2016-07-30 22:55`.
- imgDateTimeEnd en time, same as above.
- imgObject (text) target, unique identifier for object as indicated in the astronomical object table (`FOREIGN KEY`).
- imgNotes (text) imaging notes, free text.
- imgFlats (integer) number of flats
- imgDarks (integer) number of darks
- imgBias (integer) number of bias
- imgLights (integer) number of images
- imgInstrument (text) unique identifier to instrument, as indicated in the auxiliary instrument table (`FOREIGN KEY`)

- imgMount (text) unique identifier to mount, as indicated in the auxiliary mount table (`FOREIGN KEY`)
- imgCamera (text) unique identifier to camera, as indicated in the auxiliary camera table (`FOREIGN KEY`)
- imgISO (integer) ISO setting for sensor sensitivity.
- imgExpTime (integer) exposure time in seconds.
- imgEyepiece (text) unique identifier for eyepiece as indicated in the auxiliary table eyepiece (`FOREIGN KEY`). This is only valid in the case of eyepiece projection.
- imgLens (text) unique identifier to lens (e.g. Barlow), as indicated in the auxiliary lens table (`FOREIGN KEY`). Valid for negative projection.
- imgFilter (text) unique identifier to filter, as indicated in the auxiliary filter table (`FOREIGN KEY`).
- imgAccessory (text) unique identifier to accessory, as indicated in the auxiliary accessory table (`FOREIGN KEY`).

Note: this table is still in early phases of development and is not seriously tested.

4.1.4 varstarlog: visual variable star estimation log table

- obsID (integer) unique identifier for the estimate, autoincrement (`PRIMARY KEY`).
- obsSession (integer) session ID, unique identifier of session in which the estimation takes place as indicated in the auxiliary session table (`FOREIGN KEY`).
- obsDateTimeLocal (text) observation time, a string with ISO 8601 format, e.g. `2016-07-30 22:55`.
- obsTargetStar (text) target star, must be a unique identifier in auxiliary table object (`FOREIGN KEY`).
- obsCompStar1 (real) magnitude of first comparison star.

- obsCompStar2 (real) magnitude of second comparison star.
- obsTargetStarEstimate (real) magnitude estimate for target star.
- obsTargetStarEstimateError (real) error in magnitude estimate for target star.
- obsFainterThan (boolean) if the result is not a magnitude, but a *fainter than* estimation.
- obsEstimationMethod (text) estimation method, e.g. fractional, step.
- obsAAVSOchart (text) AAVSO chart identifier code.
- obsAAVSOcomment (text) comment code according to auxiliary table AAVSOCommentCode (`FOREIGN KEY`).
- obsNotes (text) free text.
- obsInstrument (text) unique identifier to instrument, as indicated in the auxiliary instrument table (`FOREIGN KEY`).
- obsMount (text) unique identifier to mount, as indicated in the auxiliary mount table (`FOREIGN KEY`)
- obsEyepiece (text) unique identifier for eyepiece as indicated in the auxiliary table eyepiece (`FOREIGN KEY`).
- obsLens (text) unique identifier to lens (e.g. Barlow), as indicated in the auxiliary lens table (`FOREIGN KEY`).
- obsFilter (text) unique identifier to filter, as indicated in the auxiliary filter table (`FOREIGN KEY`).
- obsAccessory (text) unique identifier to accessory, as indicated in the auxiliary accessory table (`FOREIGN KEY`).

Note: a report to AAVSO can be created using a script that queries information from this table and casts it into a suitable report file.

4.2 Auxiliary tables

The auxiliary tables contain descriptions of objects, equipment, sites, and other lists that are convenient to keep in good order as separate tables. I tried to avoid over–bureaucratization as much as possible, but had to accept a bit of complexity in some cases in order to keep things tidy and forcing some organization. For example, the table `objectType` was created to enable categorization of objects, to list observations by type, so to force objects to belong to only these categories. The tables related to instruments and other physical equipment have fields for tracking the beginning and end of their lives with the owner. Their purpose is to keep track of the equipment used, these are are not mandatory but can be useful. Filling this lists at the beginning might feel like a hard task, but this is only done once and I believe the results are worth the effort.

4.2.1 site

- siteID (text) unique identifier for site (`PRIMARY KEY`).
- siteLongitudeDG (real) referred to the WGS84 ellipsoid, which are the coordinates given by GPS.
- siteLatitudeDG same as above.
- siteHeightMT (real) site altitude, in meters above sea level.
- siteTimeZone (real) time difference with GMT.
- siteLightPollutionBortle (integer) estimated or measured general light pollution according to the Bortle scale.
- siteLightPollutionLocal (text) verbal description of local light pollution sources.
- siteSafety (integer) subjective evaluation of site safety. I use a scale of 1 to 10, from poor to excellent.
- siteTranquility (integer) same as above
- siteNotes (text) free text.

4.2.2 instrument

- instrumentID (text) unique identifier for instrument (`PRIMARY KEY`).
- instrumentType (text), telescope, binocular.
- instrumentBrand (text)
- instrumentApertureMM (real)
- instrumentFocalLengthMM (real)
- instrumentNotes (text)
- beginLifeSpanDate (text), a date as ISO 8601, for registering the acquisition.
- beginLifeSpanNote (text)
- endLifeSpanDate (text), same for sell off, etc.
- endLifeSpanNote (text)

4.2.3 mount

- mountID (text) unique identifier for mount (`PRIMARY KEY`).
- mountType (text), equatorial, azimuthal
- mountBrand (text)
- mountNotes (text)
- beginLifeSpanDate (text)
- beginLifeSpanNote (text)
- endLifeSpanDate (text)
- endLifeSpanNote (text)

4.2.4 eyepiece

- eyepieceID (text) unique identifier for eyepiece (`PRIMARY KEY`).
- eyepieceType (text)
- eyepieceBrand (text)
- eyepieceModel (text)
- eyepieceFocalLengthMM (real)
- eyepieceAfovDG (real) apparent field of view
- eyepieceEyeReliefMM (real)
- eyepieceFieldStopMM (real)
- eyepieceBarrelSizeIN (real) inches instead of millimeters because of convention
- eyepieceBarrelThread (boolean) does eyepiece have a filter thread?
- eyepieceNotes (text)
- beginLifeSpanDate (text)
- beginLifeSpanNote (text)
- endLifeSpanDate (text)
- endLifeSpanNote text

4.2.5 lens

- lensID (text) unique identifier for lens (`PRIMARY KEY`).
- lensType (text), barlow, coma corrector, etc.
- lensBrand (text)
- lensAmplificationFactor (real)
- lensBarrelSizeIN (real)
- lensNotes (text)

- beginLifeSpanDate (text)
- beginLifeSpanNote (text)
- endLifeSpanDate (text)
- endLifeSpanNote text

4.2.6 filter

- filterID (text) unique identifier for filer (`PRIMARY KEY`).
- filterType (text)
- filterBrand (text)
- filterBarrelSizeIN (real)
- filterNotes (text)
- beginLifeSpanDate (text)
- beginLifeSpanNote (text)
- endLifeSpanDate (text)
- endLifeSpanNote text

4.2.7 accessory

- accessoryID (text) (`PRIMARY KEY`).
- accessoryType (text)
- accessoryBrand (text)
- accessoryFunction (text)
- accessoryTo (text) unique identifier of instrument, camera, etc
- accessoryNotes (text)
- beginLifeSpanDate (text)
- beginLifeSpanNote (text)
- endLifeSpanDate (text)
- endLifeSpanNote text

4.2.8 camera

- cameraID (text) unique identifier (`PRIMARY KEY`).
- cameraType (text)
- cameraBrand (text)
- cameraModel (text)
- cameraSensorType (text)
- cameraSensorSizeXpx (integer) sensor size in pixels in the X direction.
- cameraSensorSizeYpx (integer) idem but in the Y direction.
- cameraSensorSizeXmm (real) sensor size in millimeters in the X direction.
- cameraSensorSizeYmm (real) idem in the Y direction.
- cameraPixelSizeXmicroM (real) pixel size in μm in the X direction.
- cameraPixelSizeYmicroM (real) idem in the Y direction
- cameraNotes (text)
- beginLifeSpanDate (text)
- beginLifeSpanNote (text)
- endLifeSpanDate (text)
- endLifeSpanNote text

4.2.9 object: astronomical object table

- objectMyName (text) unique identifier to object, the name by which I call this object (`PRIMARY KEY`)
- objectCommonName (text) most common name of the object
- objectType (text) type of object, according to objectType table (`FOREIGN KEY`)

- objectConstellation (text) constellation, according to constellation table (`FOREIGN KEY`)
- objectBayer (text) name in Bayer designation, if any, for stars and star–like objects
- objectFlamsteed (integer) idem for Flamsteed number
- objectHipparcos (text) idem for Hipparcos catalog
- objectAAVSO (text) idem for AAVSO unique designations
- objectVarStarType (text) for variable stars, variability type, according to The International Variable Star Index
- objectMessier (integer) number in Messier catalogue
- objectNGC (integer) idem for NGC
- objectIC (integer) idem for IC
- objectCaldwell (integer) idem for Caldwell
- objectCollinder (integer) idem for Collinder
- objectMelotte (integer) idem for Melotte
- objectOtherCatalogName (text) name of other catalog, if applicable
- objectOtherCatalogDsgn (text) designation in other catalog
- objectWiki (text) link to Wikipedia article of object
- objectNotes (text)

Note: The table suits a wide variety of objects and is only intended to put all objects in one place. I use a link to a Wikipedia article instead of adding even more information.

4.2.10 objectType

- objectType unique identifier, a two–letter code (`PRIMARY KEY`)
- objectTypeDescription (text) normally a Wikipedia page.

- objectTypeNotes text

Note: the current objectType table looks like this:

```
sqlite> select * from objectType;
objectType|objectTypeDescription|objectTypeNotes
        PL|Planet|https://en.wikipedia.org/wiki/Planet
        DP|Dwarf planet|https://en.wikipedia.org/wiki/Dwarf_planet
        GC|Globular cluster|https://en.wikipedia.org/wiki/Globular_cluster
        OC|Open cluster|https://en.wikipedia.org/wiki/Open_cluster
        NB|Nebula|https://en.wikipedia.org/wiki/Nebula
        ST|Star|https://en.wikipedia.org/wiki/Star
        CO|Comet|https://en.wikipedia.org/wiki/Comet
        NS|Natural satellite|https://en.wikipedia.org/wiki/Natural_satellite
        GX|Galaxy|https://en.wikipedia.org/wiki/Galaxy
        AS|Asteroid|https://en.wikipedia.org/wiki/Asteroid
        MT|Meteor|https://en.wikipedia.org/wiki/Meteoroid
```

4.2.11 constellation

- constellationID (text) unique identifier, abbreviation (`PRIMARY KEY`).
- constellationFullName (text) full name
- constellationGenitive (text) genitive form
- constellationWiki (text) link to Wikipedia
- constellationNotes text

This table is now included in the sample database. The first rows look like this:

```
sqlite> select * from constellation;
constellationID|constellationFullName|constellationGenitive|constellationWiki
And|Andromeda|Andromedae|https://en.wikipedia.org/wiki/Andromeda_(constellation)
Ant|Antlia|Antliae|https://en.wikipedia.org/wiki/Antlia
Aps|Apus|Apodis|https://en.wikipedia.org/wiki/Apus
Aqr|Aquarius|Aquarii|https://en.wikipedia.org/wiki/Aquarius_(constellation)
Aql|Aquila|Aquilae|https://en.wikipedia.org/wiki/Aquila_(constellation)
```

```
Ara|Ara|Arae|https://en.wikipedia.org/wiki/Ara_(constellation)
Ari|Aries|Arietis|https://en.wikipedia.org/wiki/Aries_(constellation)
Aur|Auriga|Aurigae|https://en.wikipedia.org/wiki/Auriga_(constellation)
Boo|Boötes|Boötis|https://en.wikipedia.org/wiki/Bo%C3%B6tes
```

Note: this is useful to enforce the use of correct abbreviations and genitives when creating reports.

4.2.12 AAVSO comment codes

- codeID (text) the code itself, (`PRIMARY KEY`).
- codeMeaning (text)
- codeNotes text

Note: I created a table with the comment codes that the AAVSO requires in the variable star reports. My table looks like this:

```
sqlite> select * from varstarAAVSOCommentCode;
codeID|codeMeaning|codeNotes
     B|Sky is bright, moon, twilight, light pollution, aurorae.|
     D|Unusual Activity (fading, flare, bizarre behavior, etc.)|
     I|Identification of star uncertain.|
     K|Non-AAVSO chart.|
     L|Low in the sky, near horizon, in trees, obstructed view.|
     S|Comparison sequence problem.|
     U|Clouds, dust, smoke, haze, etc.|
     V|Faint star, near observing limit, only glimpsed.|
     W|Poor seeing.|
     Y|Outburst.|
     Z|Magnitude of star uncertain.|
```

Source: AAVSO, format for reporting visual observations.

5 Use

5.1 Files

The following files form this release of *obslog*:

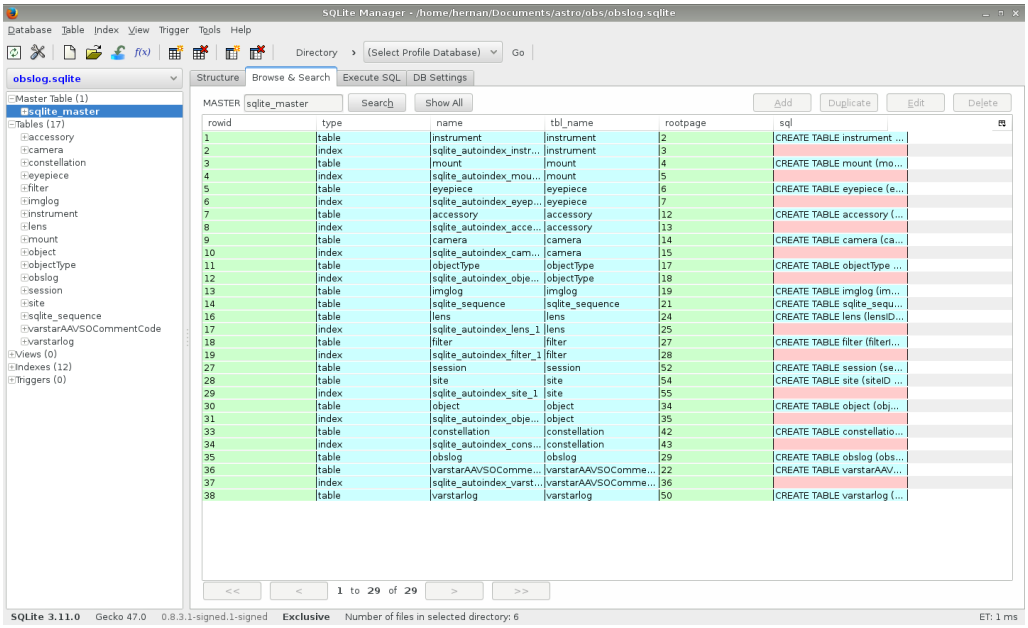

Figure 1: The obslog master table opened in Firefox SQLite Manager

.

1. this manual
2. the file `obslog_create-tables.sql` with the SQL code
3. a sample SQLite database `obslog.sqlite` generated using the SQL above

5.2 Using the precreated database

To use the packaged version of *obslog* database simple add the *SQLite manager* add–on to a Firefox browser. The add–on will be available in the options menu. You will be able to start entering data directly, see Figure 1.

It may also be possible to use any other database editor, like DB Browser for SQLite.

5.3 Creating a database

You will need to install sqlite in your system. Open a terminal, change directory to where the database will reside. Copy the file `obslog_create-tables.sql` to this directory. In the terminal, start the sqlite command dialogue by invoking the name of the file to be created:

```
sqlite3 obslog.sqlite
```

Then execute the SQL file:

```
.read obslog_create-tables.sql
```

You can check that the tables have been created by running the `schema` command:

```
.schema
```

5.4 Populating the tables

To start using *obslog* begin by filling the auxiliary tables with the details of your observing sites (`site` table), instruments (`instrument` table), eyepieces (`eyepiece` table), etc.

Once you have done this you might continue with creating a session, in the `session` table, and then creating your observations for that session in the `obslog`, `varstarlog` or `imglog` tables.

5.5 Making a database dump as backup

You can dump the whole database into a text file. This is useful as backup and also as a platform for loading the whole database again after introducing changes. Simply change the output to a file, instead of stdout, and then make a dump:

```
.output dump.sql
.dump
```

Then return the output device to stdout:

```
.output stdout
```

5.6 Examples of query

- **List all observed Messier objects**

```
SELECT DISTINCT object.objectMessier AS "OBJECT NUMBER",
                object.objectCommonName AS "OBJECT NAME"
       FROM object JOIN obslog
       ON obslog.obsObject = object.objectCommonName
       WHERE objectMessier != "NULL" ORDER BY object.objectMessier;

OBJECT NUMBER|OBJECT NAME
            2|M2
           15|M15
           27|Dumbbell Nebula
           31|Andromeda Galaxy
           33|Triangulum Galaxy
           52|M52
           56|M56
```

- **List all planetary observations**

```
SELECT
obslog.obsDateTimeLocal, object.objectType, object.objectCommonName
 FROM object JOIN obslog
 ON obslog.obsObject = object.objectCommonName
 WHERE object.objectType = 'PL'
 ORDER BY obslog.obsDateTimeLocal;

obsDateTimeLocal|objectType|objectCommonName
2010-08-19 23:40|PL|Jupiter
2010-08-22 22:55|PL|Jupiter
2010-08-22 23:00|PL|Uranus
2010-09-05 00:05|PL|Jupiter
2010-09-05 00:10|PL|Uranus
2015-08-15 02:30|PL|Uranus
```

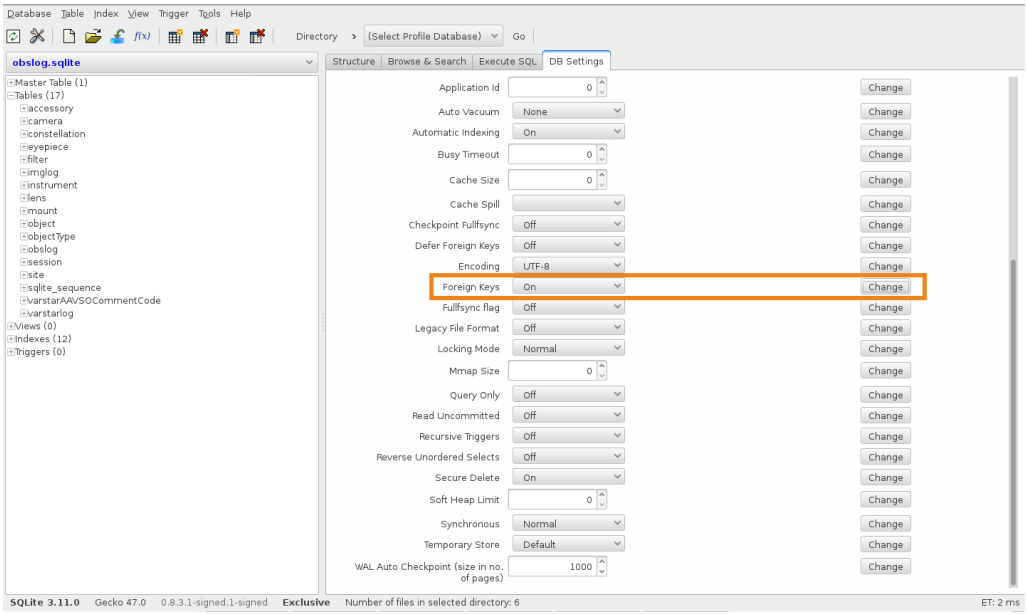

Figure 2: Activating the `FOREIGN KEY` pragma in Firefox SQLite Manager

5.7 Caveats

In its current implementation in SQLite 3 every time you start using *obslog* you need to tell SQLite to enforce the `FOREIGN KEY` pragma. This is a "feature" of SQLite 3 to ensure backwards compatibility. This can be done in two ways:

1. if you are using *obslog* in the command line, write the following command when you start working:
   `PRAGMA foreign_keys = ON;`

2. if you are using SQLite Manager in Firefox, activate the `FOREIGN KEY` pragma by changing the corresponding option in the DB Settings tab. Change the option to `ON` and then click on `Change`. See Figure 2.

6 Ideas for the future

obslog is in its infancy and is understandably limited, although I already feel very comfortable using it to store my observations. Perhaps the most acute need right now is a better collection of example queries. In some future

it might be also useful to create a graphic user interface, but I am almost certain that I will not be doing it.

7 Version history and change log

- 2016-12-16 completed the constellation table, now included in the sample database, updated manual.
- 2016-09-18 improved tables, changed transparency scale, removed Julian date as value to be inserted, updated manual.
- 2016-07-31 first release.

8 License

9 Contact

If you have suggestions and comments, please feel free to use any of these channels:

- comment in my astroblog Epistulae Astronomicae.
- send me a message in the forum Stargazers Lounge or Cloudy Nights, where my user name is Cinco Sauces.