Data mining, SourceList and AssociateList exercises

November 22, 2005

Investigating twilight behaviour from RawTwilightFlatFrames

Question 1: Find all raw twilight flat frames for the chip with name `ccd52` and the filter with `#843`, which have image statistics such that the median is larger than 1000. Only select those that have their `quality_flags` set to zero. Assign the *query* to a Python variable.
How many raw twilight flat frames satisfy the above criteria?

Solution:

```
awe> qccd52 = RawTwilightFlatFrame.chip.name     == 'ccd52'
awe> q843   = RawTwilightFlatFrame.filter.name   == '#843'
awe> qim    = RawTwilightFlatFrame.imstat.median >  1000
awe> qqf    = RawTwilightFlatFrame.quality_flags == 0
awe> query  = qccd52 & q843 & qim & qqf
awe> print len(query)
1033
```

Question 2: Using the result from the previous question, assign the MJD_OBS (the modified Julian date) of each RawTwilightFlatFrame to variable `t` and the median of the image statistics divided by the EXPTIME of each RawTwilightFlatFrame to variable `y`. Make a scatter plot of `t` versus `y`.
Solution:

```
awe> t = [rtf.MJD_OBS for rtf in query]
awe> y = [rtf.imstat.median/rtf.EXPTIME for rtf in query]
or, quicker,
awe> result = [(rtf.MJD_OBS, rtf.imstat.median/rtf.EXPTIME) for rtf in query]
awe> t, y = zip(*result)
awe> scatter(t, y)
```

Question 3: Since there seems to be a recurring pattern in the previous plot, we'll have a closer look. Take the fractional part of the `t` variable, which contains the modified Julian date for each RawTwilightFlatFrame and assign it to the variable `tfrac`. Clear the figure and make a scatter plot of `tfrac` versus `y`.
Solution:

```
awe> tfrac = [k - floor(k) for k in t]
awe> clf()
awe> scatter(tfrac, y)
```

Bonus question: Determine the time difference between Greenwich and the place where the RawTwilightFlatFrames were observed - La Silla. Does this correspond to the longitude of La Silla?

Looking at skybrightness using ReducedScienceFrames

Question 1: Plot the skybrightness as function of date.

SourceList and AssociateList

Question 1: Most SourceLists are derived from frames or images. Some however are not. Find all SourceLists which do not have a frame associated with it.
Solution:

```
awe> sls = SourceList.name != ''
awe> for sl in sls:
...   if (not sl.frame): print sl.name, sl.number_of_sources
...
```

You will have found the only lists which don't have a history and they are the standard catalogues USNO=A2.0 and SDSS-DR4. The two sourcelists are in principle growing sourcelist which will fetch the information on demand from remote servers.
Question 2: Find out the piece of sky covered by the USNO catalogue in the database.
Solution:

```
awe> usno = (SourceList.name == 'USNO-A2.0')[0]
awe> RA, DEC = usno.sources.RA, usno.sources.DEC
awe> print max(RA), min(DEC), min(RA), max(DEC)
```

This is the current coverage as of 11/11/2005 15:21 of the USNO-A2.0 catalogue in the astro-wise database.
Question 3: Find the B and R mags of the sources in the USNO catalog which are inside a distance of 0.1 degrees of position 12.0^o -29.0^o.
Solution:

```
awe> print usno.info()
awe> attrs = { 'RA': [], 'DEC': [], 'USNO_RMAG': [], 'USNO_BMAG': [] }
awe> area = (12.0, -29.0, 0.1)
awe> r = usno.sources.area_search(Area= area, dict=attrs)
awe> nos = len(r[usno.SLID])
awe> for k in range(nos):
...   print attrs['USNO_BMAG'][k], attrs['USNO_RMAG'][k]
...
```

Question 4:
AssociateLists are used for the Global Astrometric Solution. We are going to inspect some of these lists in the database. These are easily found since there names all start with GAS:

```
awe> als = AssociateList.name.like('GAS*')
awe> len(als)
```

Take one of these Associatelists, for example:

```
awe> al = als[2]
```

and find out the sourcelists they have been made from and find out the pointing and corresponding CCD's.
Solution:

```
awe> sls = al.sourcelists
awe> for sl in sls:
...   if (sl.frame):
...    print sl.frame.chip.name, sl.frame.astrom.CRVAL1, sl.frame.astrom.CRVAL2
```