Handy Queries

This project is proceeding from two points of view:

1. Integrate the spectral libraries from astronomical projects.
2. Take in spectra from large and small telescope programs.

The first letter in FITS is flexible, and to that end it is necessary to bring in the stardard star *fits* files headers. The vocabulary of the originating observatories and small telescope observers differ. Databases require exact matches, and to these ends require a process to assure data integrity and referential ingegrity. The astrometric positions are not precise, and 'neighborhood' (cone) are required. Target names have various assumptions built in: e.g.: occasional use of two spaces rather than one. While the brain can accomodate the difference – machines trip over the mis-step.

Astropy was used to open files, iterate over one or more HDUs (Header Data Unit) consists of **'CARDS'** with a keyword, value and optional comment. The HDU is scanned and converted into a Python dictionary by the 'fits2psqlraw' python tool. The JSON keyword is the FITS keyword, the JSON value field is an embedded dictionary with the 'value' and 'comment' part. The 'fits2psqlraw' python tool produces output to 'stdout' (redirect to a file) and is loaded into PostgreSQL as a 'raw' file. The 'raw' file can be accessed (think Materialized View) to transliterate the observatory into the database's vocabulary.

Example SELECT statement

Here external functions **r2s** and **d2s** are used to convert decimal ra and decimal Dec to traditional sexigesimal notation. The *quote_literal* places tick-marks around the text for clarity and possible use later on in the process. The header is declared to be a **jsonb** (binary json) field; the -> operator brings out fields and the ->> operator renders them into text form.

Note: This is from the NGSL original files, where the "OBJECT" is called "TARGOBJ" etc.

```
    select target, r2s(ora) as "RA", d2s(odec) as "DEC",
           quote_literal(header -> 'HDU[0]' -> 'GRATING' ->> 'value') as "Grating",
           quote_literal(header -> 'HDU[0]' -> 'SPECTYPE' ->> 'value') as "SpType"
    from rawngslheaders
    where header -> 'HDU[0]' -> 'SPECTYPE' ->> 'value' = 'composite'
    order by ora::integer/15,odec
    ;
```

A Python function useful to pretty print JSON entries:

```
    CREATE FUNCTION pp_json(j JSON, sort_keys BOOLEAN = TRUE, indent TEXT = '    ')
    RETURNS TEXT AS $$
      import simplejson as json
      return json.dumps(json.loads(j), sort_keys=sort_keys, indent=indent)
    $$ LANGUAGE PLPYTHONU;
```

Summary

This page will be expanded to include more examples of queries suited to the task.

Simbad

Given a list of target names, not necessarily the SIMBAD mainid names, locate them and dig out the salient data for the bigger database. This produces a **.raw** file. The main **basic** table has a **iod**, other tables link to this using the field ane **oidref**. In the case of the **ident** table, multiple star names are tied to one **oidref**, which ties back to the other tables. It is possible to move between the tables using just the **oidref**. Here ident is used to match the given target name from observer, allfluxes digs out the flux values, and basic provides the ra,dec,spectral type and qualifier.

First the target names are entered into a simple database, using a simple Comma-Separated-Value file. All that is needed is one column with target names that match alternate names within SIMBAD. SIMBAD is pretty liberal with the star names, mispellings (this example Betelguese is mispelled) but some planetarium vendors alter star names up with their own decisions. Leading zeros may be omitted when SIMBAD requires them and minus signs are changed to colons or worse. Latex tends to use typesetting characters in lieu of the ASCII used within SIMBAD.

```
SELECT  ident.oidref, ident.id, basic.otype_txt,
        basic.ra, basic.dec, basic.sp_type, basic.sp_qual,
        allfluxes.*, alltypes.*
FROM ident, TAP_UPLOAD.giventargets
join allfluxes on   ident.oidref = allfluxes.oidref
join alltypes  on   ident.oidref = alltypes.oidref
join basic     on   ident.oidref = basic.oid
where giventargets.starname = ident.id;
;
```

```
 select b.id as "Given Target",
        header -> 'TARGNAME' ->> 'value' as "NGSLNAME",
        fqpname,r2s(b.ora)  as "RA",
        d2s(b.odec) as "DEC"

 from stis           a
 join rawgiventargets b
 on q3c_join(a.ora,a.odec, b.ora,b.odec, 10.0)
 order by (b.ora::int)/15, b.odec
 ;
```

```
1
```

The original list from a email was used to produce, by hand and editor a simple list. The gratitious column about the payload was added. This is then **Load -> table database** into TOPCAT extended with the proper jdbc package. PostgreSQL is used:

```
 DROP TABLE    IF EXISTS giventargets;
 DROP SEQUENCE IF EXISTS giventargets_sequence;
 CREATE SEQUENCE         giventargets_sequence START 100000;

 CREATE TABLE giventargets (
    uniqueid  integer PRIMARY KEY DEFAULT nextval('giventargets_sequence'),
    starname  text,
    program   text
 );

 insert into giventargets (starname,program ) values
    ('HD123299'       ,  'Lowspec3 Miles' ),
    ('HD142373'       ,  'Lowspec3 Miles' ),
    ('HD142908'       ,  'Lowspec3 Miles' ),
    ('HD155763'       ,  'Lowspec3 Miles' ),
    ('HD164058'       ,  'Lowspec3 Miles' ),
    ('HD160762'       ,  'Lowspec3 Miles' ),
```

```
18        ('HIP 66700'      ,  'Lowspec3 target' ),
19        ('HD 108907'      ,  'Lowspec3 target' ),
20        ('HD 142926'      ,  'Lowspec3 target' ),
21        ('HD 138629'      ,  'Lowspec3 target' ),
22        ('HIP 95413'      ,  'Lowspec3 target' ),
23        ('HD 166014'      ,  'Lowspec3 target' ),
24        ('HD145976'       ,  'UVEX3 Miles' ),
25        ('HD038545'       ,  'UVEX3 Miles' ),
26        ('HD142373'       ,  'UVEX3 Miles' ),
27        ('HD142807'       ,  'UVEX3 Miles' ),
28        ('4Her'           ,  'UVEX3 target' ),
29        ('Betelguese'     ,  'UVEX3 target' ),
30        ('Procyon'        ,  'UVEX3 target' ),
31        ('tetCrB'         ,  'UVEX3 target' ),
32        ('HD155763'       ,  'Tragos3 Miles' ),
33        ('HD142373'       ,  'Tragos3 Miles' ),
34        ('HD136064'       ,  'Tragos3 Miles' ),
35        ('eta Dra'        ,  'Tragos3 target' ),
36        ('4 Her'          ,  'Tragos3 target' ),
37        ('53 Boo'         ,  'Tragos3 target' ),
38        ('AGDra'          ,  'Tragos3 target' );
```

copy (select * from giventargets) TO ‘/tmp/giventargets.csv’ WITH CSV DELIMITER ‘,’ HEADER;

This appears as a rather complicated name in the **Table list** of TOPCAT. The table name can then be changed to something easy to type for an **UPLOAD** key for joins using the TOPCAT TAP interface to the SIMBAD service.