天文数据:操作、归档与发布

Release 0.5

何勃亮

2013 年 12 月 13 日

Contents

I 系统安装 **1**

1 CentOS **5**
1.1 软件源 5
1.2 安装基本包 5
1.3 天文软件环境 6
1.4 数据库 7

2 Ubuntu **9**

3 Mac OS **11**
3.1 安装 `Xcode` 11
3.2 `Homebrew` 11

4 Python **13**

5 Aladin & Topcat **15**
5.1 Java 15
5.2 Aladin 15
5.3 Topcat 15

II 基础知识 **17**

6 天文数据 **19**
6.1 FITS 19
6.2 VOTable 19

7 Python **25**
7.1 NumPy 25

7.2 Pandas . . . 25
7.3 Astropy . . . 26
7.4 IPython . . . 27

8 PostgreSQL **33**
8.1 数据类型 . . . 33
8.2 基本命令 . . . 34
8.3 SQL 指令 . . . 41
8.4 pgAdmin . . . 50
8.5 pgSphere . . . 51

III 玩转数据 **55**

9 文本文件 **57**
9.1 基础知识 . . . 57
9.2 查看文件 . . . 57
9.3 筛选文件 . . . 58
9.4 文本替换 . . . 61
9.5 文件排序 . . . 61
9.6 文件统计 . . . 64
9.7 pandas . . . 65

10 FITS 文件 **67**
10.1 FITS 文件校验 . . . 67
10.2 Funtools . . . 68
10.3 Python FITS 操作 . . . 74

11 二进制文件 **75**

IV 数据归档 **79**

12 简介 **81**
12.1 什么是数据归档? . . . 81
12.2 数据归档 . . . 81
12.3 元数据 . . . 83

13 数据库 **85**
13.1 准备工作 . . . 85
13.2 建立数据库 . . . 85
13.3 建表 . . . 86
13.4 数据录入 . . . 88
13.5 建立索引 . . . 88

14 数据传输 **91**

14.1 基础 . . . 91
14.2 案例 . . . 93

V 数据接口 95

15 访问数据 97
15.1 psql 访问数据库 . . . 97
15.2 Python 访问数据库 . . . 98
15.3 数据下载 . . . 99

16 在线数据访问接口 103
16.1 CDSClient . . . 103
16.2 astroquery . . . 113
16.3 SDSS DR10 . . . 114

VI VO 程序 115

17 Aladin 117
17.1 使用案例 . . . 118

18 Topcat 125

Bibliography 133

Part I

系统安装

工作环境基于 **Linux** 或者 **Mac OS** ,安装的软件包括:

- 程序编译环境: gcc、g++、gfortran、make、cmake、Java
- 软件库:cfitsio、funtools、cdsclient
- 命令行工具:awk、sed、wc、sort

CHAPTER 1

CentOS

CentOS [1] 是 *社区企业操作系统* (Community ENTerprise Operating System) 的缩写,它是 Redhat Enterprise Linux (RHEL) 的源码再编译,拥有与 RHEL 相同的性能和完全兼容,是服务器端操作系统的首选之一。与 CentOS 类似的系统还有 Scientific Linux [2] 、Oracle Linux [3] 等。

以最新的 CentOS 6.5 为例,安装软件,首先最小化安装一个 Linux。

1.1 软件源

除自带的源外,再添加 EPEL (Extra Packages for Enterprise Linux) 软件源

```
rpm -ivh http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
```

1.2 安装基本包

系统更新

```
yum -y update
```

开发包

```
yum -y install openssh-clients vim gcc gcc-c++ gcc-gfortran git wget
redhat-lsb-core asciidoc xmlto make cmake autoconf unzip gdb  bison
```

开发库

```
yum -y install zlib-devel.x86_64 libyaml-devel.x86_64 PyYAML.x86_64
libyaml.x86_64  perl-YAML.noarch libedit-devel libffi-devel  readline-devel
```

[1] http://www.centos.org/,软件下载可选取国内的多个源,比如:http://mirrors.163.com/centos/。
[2] http://scientificlinux.org/
[3] http://linux.oracle.com/

```
libxml2-devel libxslt-devel flex libicu-devel openssl-devel.x86_64
tk.x86_64 tix-devel.x86_64 tk-devel.x86_64 libcurl-devel.x86_64
expat-devel.x86_64 expat.x86_64 perl-Time* pam.x86_64  pam-devel.x86_64
pcre-devel.x86_64 pcre.x86_64 gd-devel
sqlite-devel.x86_64 sqlite.x86_64 bzip2-devel.x86_64 db4-devel.x86_64
ImageMagick-devel.x86_64 ncurses-devel gdbm-devel
```

计算与天文库

```
yum -y install octave-devel.x86_64 octave-image.x86_64 octave.x86_64 gsl-devel
gsl gnuplot cfitsio-devel.x86_64  cfitsio.x86_64  funtools.x86_64  wcslib.x86_64
wcstools.x86_64
```

1.3 天文软件环境

1.3.1 Python

Python CentOS 默认的 python 版本是 2.6,下面安装最新的 2.7.6。

```
tar xjvf Python-2.7.6.tar.bz2
./configure --prefix=/usr/local --enable-shared
make && make install
```

1.3.2 CDSClient

下载安装 Mac OS 可以直接使用 brew 安装

```
wget -c http://cdsarc.u-strasbg.fr/ftp/pub/sw/cdsclient.tar.gz
tar xzvf cdsclient.tar.gz
./configure --prefix=/usr/local
make && make install
```

1.3.3 Funtools

下载安装

```
wget -c https://www.cfa.harvard.edu/~john/funtools/
tar xzvf funtools.tar.gz
./configure --prefix=/usr/local
make && make install
```

1.4 数据库

PostgreSQL [4]

下载

```
wget http://ftp.postgresql.org/pub/source/v9.3.2/postgresql-9.3.2.tar.gz
tar xzvf postgresql-9.3.2.tar.gz
```

编译安装

```
./configure --prefix=/usr/local
make
make install
cd contrib
make
make install
```

安装 pgsphere

请在培训网站下载:http://code.china-vo.org/tn/astrodb2013/pgsphere-1.1.1.tar.gz

```
tar xzvf pgsphere-1.1.1.tar.gz
make USE_PGXS=1 PG_CONFIG=/usr/local/bin/pg_config
make USE_PGXS=1 PG_CONFIG=/usr/local/bin/pg_config install
```

[4] http://www.postgresql.org

CHAPTER 2

Ubuntu

Ubuntu 是当前非常流行的桌面端 Linux 发行版本。下面以 Ubuntu 13.10 为例安装相关的软件。

更新系统

```
sudo apt-get update
sudo apt-get upgrade
```

开发包

```
apt-get install gcc* build-essential gfortran
```

开发库

```
apt-get install python-dev bison libzip-dev bzip2
libgdbm-dev  libgdbm3 libpcre3-dev libcurl4-openssl-dev
sqlite3 libxml2-dev
```

计算与天文库

```
apt-get install libcfitsio3-dev funtools gnuplot libgsl0-dev
libwcs4 libwcstools-dev wcslib-dev
```

数据库

```
apt-get install postgresql postgresql-contrib
postgresql-server-dev-all
```

Java

```
apt-get install openjdk-7-jdk
```

CHAPTER 3

Mac OS

Mac OS 是苹果公司发布的安装于苹果电脑上的操作系统,它的历史比较曲折,足以写上一本书。但是它主要还是基于 FreeBSD 系统,也就是一个 Unix 的变种,Mac OS 系统在科学家中使用的也相当的多。下面介绍一个在 Mac OS 环境下安装科学相关软件的方法和步骤。

Mac OS 环境下安装软件有多个 Fink [1] 、Macports [2] 以及 Homebrew [3] ，下面着重介绍 Homebrew 。

3.1 安装 Xcode

用户需要登录 Mac OS 的 App Store 账户,安装 Xcode ,安装完毕后,打开 Xcode 安装命令行工具,这是后面所有工作的基础,因为 Xcode 包含基本的编译软件:cc, c++, git, make 等。

3.2 Homebrew

安装

```
ruby -e "$(curl -fsSL https://raw.github.com/mxcl/homebrew/go/install)"
```

安装完毕后,需要更新库:

```
brew update
```

Homebrew 默认安装软件的路径是 /usr/local/Cellar/prog/versioin ,以 CDSClient 为例,它的安装路径就是 /usr/local/Cellar/cdsclient/3.71 ,然后其可执行程序会以软链接方式接入 /usr/local/bin ,这是默认在用户 PATH 下的,可以直接在命令行使用。Homebrew 以 git 管理所有的软件安装脚本。

[1] http://www.finkproject.org/
[2] http://www.macports.org/
[3] http://brew.sh/

科学库 Homebrew 派生了多个专门的子项目来归档软件源。Homebrew-science [4] 是科学类软件的汇集地。

```
brew tap homebrew/science
brew update
```

3.2.1 brew 命令

```
brew update     # 更新安装脚本
brew upgrade    # 更新已安装的软件
brew install prog       # 安装 prog 软件
brew search pattern     # 搜索 pattern 软件
```

3.2.2 安装软件

开发库

```
brew install wget
brew install gfortran
brew install python
```

数据库

```
brew install postgresql
```

计算与天文库

```
brew install gnuplot cfitsio cdsclient gsl
```

3.2.3 其他

Funtools

```
wget -c https://www.cfa.harvard.edu/~john/funtools/
tar xzvf funtools.tar.gz
./configure --prefix=/usr/local
make && make install
```

[4] https://github.com/Homebrew/homebrew-science

CHAPTER 4

Python

如前,安装好 python 基本环境后,需要逐步安装需要的软件。

安装 Distribute

```
wget -c https://pypi.python.org/packages/source/d/distribute/distribute-0.6.49.tar.gz
tar xzvf distribute-0.6.49.tar.gz
python setup.py install
```

安装 pip

```
wget -c https://raw.github.com/pypa/pip/master/contrib/get-pip.py
python get-pip.py
```

安装科学计算库

```
pip install numpy
pip install scipy
pip install matplotlab
pip install pyfits
pip install astropy
pip install psycopg2
pip install pandas
pip install requests
pip install astroquery
```

安装 ipython

```
pip install tornado
pip install ipython
```

CHAPTER 5

Aladin & Topcat

Aladin 和 Topcat 均是 Java 应用程序,因此需要先安装 Java 运行环境

5.1 Java

下载 JDK 安装包:

```
http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html
```

- CentOS 用户下载安装 RPM 包。
- Ubuntu 用户可以使用 apt-get 安装 Java。
- Mac OS 用户下载 dmg 文件安装。

5.2 Aladin

Aladin 是法国斯特拉斯堡天文数据中心发布的一款 VO 软件, 最新版本 v7.5, 下载地址: http://aladin.u-strasbg.fr/

推荐下载 Jar 包:http://aladin.u-strasbg.fr/java/nph-aladin.pl?frame=get&id=Aladin.jar

程序运行

```
java -jar Aladin.jar
```

Mac OS 用户也可以下载 dmg 包安装。

5.3 Topcat

Topcat 是 Tool for OPerations on Catalogues And Tables 的缩写,它是一个交互式的表格数据处理平台。由布里斯托尔大学物理系天体物理研究组的 Mark Taylor 博士开发维护。

主页 http://www.star.bristol.ac.uk/~mbt/topcat/ 。

下载地址: http://www.star.bristol.ac.uk/~mbt/topcat/#install

推荐下载 Jar 包:http://www.star.bristol.ac.uk/~mbt/topcat/topcat-full.jar

程序运行

```
java -jar topcat-full.jar
```

Mac OS 用户也可以下载 dmg 包安装。

Part II

基础知识

CHAPTER 6

天文数据

天文数据的主要存储格式是 FITS，数据交换格式则有 VOTable。

6.1 FITS

FITS 是传统的天文数据存储格式，可以存储图像、星表、光谱等。

6.2 VOTable

6.2.1 简介

VOTable 是一种 XML 数据格式，贯穿于所有 VO 软件和服务之间，用一种统一的方式来呈现表列数据（特别是天文星表，但也可以用于所有形式的数据交换）。采用 XML 的好处是能够充分利用行业内标准的工具和软件，易于实现互操作性，更重要的是它提供了对数据进行丰富的元数据描述的机制，比如对表列的描述甚至对整个数据体系的描述。

VOTable 是 IVOA 推出的第一个标准，它把我们熟悉的一些标准（比如 FITS）的优点与对高效表列数据交换的需求结合了起来。一方面，VOTable 支持把数据嵌入在文档自身之中以满足小表数据的高效交换；同时也支持远程化和数据流以适应网格计算的环境。虽然 VOTable 和 FITS 的二进制表格格式有许多非常相似的地方，但 VOTable 直接继承于 Astrores1（由 CDS 开发）和可扩展科学交换语言 2（Extensible Scientific Interchange Language，XSIL，由加州理工学院开发）。实际上，尽管人们有意地把从 FITS 二进制表向 VOTable 转换的过程设计为一种可逆操作，但使用所有 VOTable 特性生成的新文档可能还是无法无损地转换为 FITS 表。

每个数据表称为一个资源（resource）。VOTable 格式可以实现包含多个资源的整个数据集的交换。不过，在实际操作中绝大多数的 VO 服务对于一个查询的响应只是一个单一资源。在本章中我们要讨论 VOTable 的文档格式，VOTable 的一般用途以及它的可能用途，最终达到更好地理解该格式的目的。

6.2.2 VOTable 文档格式:概述

VOTable 的正式规范由 IVOA 文档库（http:// www.ivoa.net/ Documents/ latest/ VOT.html）负责维护，我们这里只是简单地介绍它的特点和优势，讨论它在 VO 中当前的应用。一个 VOTable 文档的关键特征包括：

元数据的完整表述：描述性的元素遍布于整个 VOTable 文档、每个资源、表中的每一列、每个数据组（相关联的列或者整个资源）。VOTable 中元数据实体的信息可以被较低层次的 VOTable 元素继承（比如一个 <PARAM> 元素适用于一个 <RESOURCE> 内的所有元素），实体的属性对实体本身提供了更具体的描述（例如 units（单位）、format（格式）、或者 status codes（状态码）等）。

资源的层次结构：这样就允许一个数据服务可以从其管理域内的多个数据集返回结果（例如把一个天文台的数据集按照巡天项目和观测申请分组返回），同时一个计算服务可以为某个操作返回多个表（例如一个星系分类服务可以把一幅图像中探测到的所有的源编入一个 RESOURCE 中作为一个表返回，而把在该图像上剪切得到的星系图像的访问链接放在另外一个表中）。

每个资源内数据和元数据的分离：从一个 VOTable 数据中可以只解析它提供的那些列，然后客户端程序还可以为从表中读取出来的一列数值合理地赋予一个有意义的标识符。元数据参数元素可能会影响到表的处理和客户端行为，例如，服务误差条件的探测或者简单地返回数据检索的查询参数。

UCD（统一内容描述符，Unified Content Descriptor） 在表列定义中的使用：UCD 为表中的一个列提供了一个唯一的标识符。这样客户程序就能够在一个表中准确地找到定义位置或者其他参数的列，而不用担心数据提供者如何为他们数据表中的列命名以及它们在表中的位置。在进行多服务查询和多 VOTable 处理时，UCD 提供了对处理结果进行有意义的和统一的逻辑并合的可能。

数据可以存放于 VOTable 文档内部，或者以远程文件和二进制流的形式存在。一个 VOTable 的 <DATA> 元素可以包含：

- **纯 XML** 表中仅包含 VOTable 规范中允许的 <TABLEDATA> 元素内的 XML 元素。
- **简单的二进制数据**一个特定单元中的数据可以封装成一个 <TABLEDATA> 元素内的一个 CDATA XML 元素，或者在一个 <BINARY> 元素内指向某个远程数据源的 <STREAM> 元素。
- **FITS 二进制表**一个数据表可以包含一个 <STREAM> 元素，用来定义一个获取 FITS 二进制表格格式文件的访问指针（access reference）。列的元数据可以作为文档的一部分包含在一个 VOTable 中。同时，VOTable 还支持对特定扩展的访问。

一个表中的个体单元可以在任何时候包含一个 CDATA 标签来在单元内定义一个经过编码处理的大型对象。目前支持的编码格式包括 gzip、base64 和动态（dynamic）（用来指定一个远程资源，编码类型由 HTTP 协议中的 MIME 头提供）。CDATA 标签最常见的是与图像访问指针（image access references）一起使用，比如一个 URL，访问它时会返回一幅图像。图像或者是静态的，或者是由拼接、剪切等服务动态生成的。

可移植性：XML 是 ASCII 码格式文档，可以方便地在不同系统间迁移，改变字节排列顺序以

及浮点数的呈现方式。虽然 VOTable 没有能力定义一套自己的、能让任何机器都可读的二进制格式(虽然原始二进制光栅数据格式可以采用,但结构性的图像数据只能用前面提到的 CDATA 标签中的动态编码方式提供),二进制数据可以用上面提到的方法来编码(像 FITS 这样的格式就具有机器无关性)。

具体到每个单独服务,它们对 VOTable 规范具有的灵活性的支持是有很大差异的。大部分的数据和计算服务只是利用了这种灵活性的一小部分。不过,通用客户软件的开发者们如果想推出一套强健的应用就应该支持 VOTable 所有的功能特性。用户在使用 VOTable 相关应用软件时需要谨慎,因为并不是 VOTable 中所有的信息都会被所用的软件支持。数据发布者在写 VOTable 的时候应该意识到并不是所有的 VOTable 解析器和客户程序都支持规范中的所有特性,在描述关键信息时应该避免使用"怪僻"的特性。

6.2.3 VOTable 示例

在我们讨论 VOTable 格式的细节之前,不妨看一段 VOTable 的实例以作参考。下面这个表是一个普通的 VO 星表服务返回的一段有代表性的结果,显示出一个单一的 RESOURCE,它包含一个两列(也就是 FIELDs)的 ASCII 表。你会注意到,通过属性来描述元素的做法对理解元数据标签(比如 PARAM 和 FIELD)的含义具有重要意义,但这并不是针对数据本身。在一些情况下,对某些特定的服务类型而言,在返回的结果中必须包括要求的属性(详细的必要元素和属性列表可以参考相关服务规范)。

```
<?xml version="1.0"?>
<VOTABLE version="1.1"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="http://www.ivoa.net/xml/VOTable/v1.1">
        <COOSYS ID="J2000" equinox="J2000." epoch="J2000."
        system="eq_FK5"/>
        <RESOURCE name="myFavoriteGalaxies">
        <TABLE name="results">
                <DESCRIPTION>Velocities and Distance estimations</DESCRIPTION>
                <PARAM name="Telescope" datatype="float"
                                ucd="phys.size;instr.tel" unit="m" value="3.6"/>
                <FIELD name="RA" ucd="pos.eq.ra;meta.main" ref="J2000"
                        datatype="float" width="6" precision="2" unit="deg"/>
                <FIELD name="Dec" ucd="pos.eq.dec;meta.main" ref="J2000"
                        datatype="float" width="6" precision="2" unit="deg"/>
                <DATA>
                        <TABLEDATA>
                        <TR><TD>010.68</TD><TD>+41.27</TD></TR>
                        <TR><TD>287.43</TD><TD>-63.85</TD></TR>
                        </TABLEDATA>
                </DATA>
        </TABLE>
        </RESOURCE>
```

```
</VOTABLE>
```

结合 VOTable 规范和上面的例子,我们发现一个 VOTable 由下列部分组成:

- **VOTable** = 元数据 + 由一组 TABLEs 构成的 TABLEDATA
- **元数据** = 一系列 PARAM + 一系列 INFO + 一系列 DESCRIPTION + 一系列 LINK + 一系列 FIELD + 一系列 GROUP
- **TABLE** = FIELD 列表 + TABLEDATA
- **TABLEDATA** = 行的序列串 (TR 标签)
- **行** = 单元列表 (TD 标签)
- **单元** = 基元 (单数、变长或多维数组)
- **基元** = 整数、字符、实数等

为了更好地了解 VOTable,下面我们把这个示例文档拆开,对每个元素进行详细地介绍。

1. VOTable 元素

这是 XML 文档树的根元素, 在一个文件中必须并且只能出现一次。在元数据部分, VOTABLE 的子元素通常包括一个或者多个 <DESCRIPTION>、<PARAM>、<INFO> 和 <COOSYS> 元素。同时, 作为我们最关心的部分, 一个表可以包含一个或者多个 <RESOURCE> 元素。

2. RESOURCE 元素

简单说来,一个 <RESOURCE> 就是一组相关联的表。<RESOURCE> 是可以迭代的,可以包含其他的 <RESOURCE> 元素,这意味着组成一个 <RESOURCE> 的相关联的一组表能够定义一个更复杂的数据结构。

一个 <RESOURCE> 元素可以有 name 或者 id 两个属性中的一个或者全部;也可以标识为 type="meta",表示这个资源只是描述性质的,在其所有的子元素中都不包含真正的数据。最后,<RESOURCE> 元素可以包含一个 utype 属性来把这个元素链接到一些外部数据模型 (这个特性在 VOTable 规范 1.1 版中被首次引入)。

3. TABLE 元素

一个 <TABLE> 元素永远会被包含在一个 <RESOURCE> 中,可能包含如下描述性的元素:

- DESCRIPTION - 关于该 <RESOURCE> 的描述文本
- FIELD – 描述表中的一列
- PARAM – 一个常数值
- GROUP – 用来为 <FIELD> 或者 <PARAM> 元素建立逻辑关联
- LINK - 指向其他 (外部) 文档或数据的指针

上面这些元素构成了一个资源的元数据;真正的数据开始于下面将要描述的 <DATA> 标签。<DATA> 元素可能的属性包括 name 和 id 以及 ref。ref 给出的是前面描述过的一个表的

id,它的出现指明这个 <TABLE> 与被参照的表具有相同的结构,这样就避免了在一个文档中多次重复一个表的完整描述信息的问题。

4. FIELD 和 PARAM 元素

一个 <FIELD> 元素是对一个表列的一段描述。它可以包含一些其他的描述性元素,比如 <DESCRIPTION>,指向外部文档或数据的 <LINK>,以及 <VALUES> 元素。<VALUES> 元素可以通过 <MIN>/<MAX> 子元素来定义一个取值范围,或者利用 <OPTION> 子元素定义一个取值列表。<FIELD> 元素的属性用来指定:

- 如果该域将在后面被引用,则可以为它指定一个 name 或者 ID;
- 值的类型、大小、数组大小、精度;
- 值的单位(使用 IVOA 指定的词汇表);
- 这个量标准的 UCD 分类;
- 一个 utype 用来指向外部数据模型;
- 一个 ref 用来在定义中引用该文档中的另外一个元素。

<PARAM> 元素和 <FIELD> 元素相仿,只不过它需要一个常数值。它和 <FIELD> 具有相同的属性集,此外它还要求有一个必需的值属性来指定该参数的值。<PARAM> 元素通常用来为 <RESOURCE> 定义一个全局数值(比如查询参数、服务的内部参数等)。<PARAM> 可以以特定的单位来定义,也可以通过属性将其和更具意义的 UCD 联系起来。

5. INFO 元素

一个 <INFO> 元素(未在上面实例中给出)是 <PARAM> 的一个限制类。它一般会给出一个信息值(例如服务返回状态、表中的行数等),支持的属性只包括 name 和 value。

6. DATA 元素

<DATA> 元素在 <TABLE> 中是唯一的(但是一个 <RESOURCE> 可能会包含多个 <TABLE> 元素)。VOTable 支持以下三种数据格式:

TABLEDATA

一个纯粹的 XML 表,用 <TR> 元素定义行,用 <TD> 元素定义单元。数据列的出现顺序必须与 <FIELD> 中定义的一致,同时一列中的所有单元必须有相同的格式。空单元用一个简单的空 XML 标签表示(例如“<TD/>”或者“<TD></TD>”)。

FITS

一个 FITS 二进制表,可以包含 extnum 属性来指向序列流中的一个特定扩展。头关键词通常在元数据部分用 <PARAM> 描述。

BINARY

用于高效传输,数据进行了编码处理(例如 base64、gzip、dynamic,使用 dynamic 时需要相关服务指定 MIME 类型)。

FITS 和 BINARY 格式必须包含一个 <STREAM> 元素,例如:

```
<TABLE>
        <FIELD name=.../>
        <DATA>
                <FITS extnum="2">
                        <STREAM encoding="gzip"
                                href="ftp://my.nvo.org/myfile.fits.gz"/>
                </FITS>
        </DATA>
</TABLE>
```

7. GROUP 元素

<GROUP> 元素（未在上面示例中给出）在 VOTable 规范 1.1 版中被引入，用来定义 <FIELD> 和 <PARAM> 元数据元素的逻辑关联。

6.2.4 在软件中读取 VOTable

解析一个 VOTable 的过程对所有程序语言而言都是大致相同的，当然由于使用的语言和 XML 解析器的不同会在细节上有些差异。本质上说，它需要从 VOTable 的根元素开始，沿着文档中的元素树“走”下去。需要注意的是有些元素是允许重复的。沿途会遇上一些元数据标签，它们还会在文档中带有子元素。这些情况需要代码能够正确处理。现在先不考虑元数据，那么一个 VOTable 读取的过程可以用如下的伪代码表示：

```
foreach (RESOURCE element) {
        foreach (TABLE within the RESOURCE) {
                foreach (FIELD/PARAM element)
                process the FIELD/PARAM and its attributes
                foreach (TR in the TABLEDATA element)
                foreach (TD in the row)
                        process the data in the cell
        }
}
```

注意，这里我们忽略了 <GROUP> 的使用，同时为简单起见只处理了 XML 格式的 <DATA> 部分。不过你会看出文档元素的层次结构和读取它们的循环结构两者之间符合的非常好。

`astropy.io.votable` 可以很好的支持 VOTable 的读写，参见：http://docs.astropy.org/en/stable/io/votable/index.html

6.2.5 CSV

文本格式文件，CSV 是以逗号隔开字段的文本数据文件。

CHAPTER 7

Python

Python 在科学计算上的应用越来越多，Python 语言的一大特点是它是一个胶水语言，语言本身的性能不高，但是众多的库是使用传统的语言 Fortran、C 等写成，通过 Python 本身的机制将其集成到 Python 库中，计算性能大幅度提高。

7.1 NumPy

NumPy 是 Python 科学计算的基础包，它提供了众多基础的模块：

- 快速高效的多维数组模型 `ndarray`
- 数据集读写工具
- 矩阵、线性代数运算、FFT、随机数生成等
- 用于将传统的 C、C++、Fortran 等代码集成到 Python 的工具

参考

- http://www.numpy.org/
- http://www.scipy.org/

7.2 Pandas

pandas 是近期发展的比较好的一个高效快捷处理结构化数据的工具集。pandas 的数据对象是 `DataFrame` ，它类似于 R 语言的 `data.frame` ，是一个面向列的二维表结构，含有行标和列标。结构类似于数据库或者 Excel 表格。

pandas 用于丰富的针对结构化数据的操作函数：

- 输入输出 (文本文件、`CSV`、`HTML`、`JSON`、`HDF5`、`SQL`、`Excel`、)
- 统计相关的函数

- 序列函数
- 集成了 Matplotlab 的绘图函数
- DataFrame
- Panel

参考

- http://pandas.pydata.org/
- 代码库 https://github.com/pydata/pandas
- 文档 http://pandas.pydata.org/pandas-docs/stable/

7.3 Astropy

Astropy Project 是近期比较热的一个天文数据处理 Python 项目，它由 2 部分组成：**astropy core package** 和 **Affiliated Package**，前者是核心通用库，后者是一系列专用的库。自 2012 年 6 月 19 日推出 0.1 版以来，已经连续发布了 6 个版本，目前的最新版本是 0.3.0。它的社区也很活跃。

一些相关网站

- Astropy Project http://www.astropy.org
- 代码库 https://github.com/astropy/astropy
- 在线文档 http://docs.astropy.org/en/stable/

7.3.1 软件包

astropy 包含了大约 16 个核心包。

- `astropy.constants` 天文和物理常数
- `astropy.unit` 单位计算
- `astropy.nddata` N 维数据集
- `astropy.table` 表格数据
- `astropy.time` 时间计算
- `astropy.coordinates` 坐标系
- `astropy.wcs` WCS
- `astropy.io` 数据读写
- `astropy.io.fits` FITS 文件读写，语法与 pyfits 相同。
- `astropy.io.ascii` 文本文件读写
- `astropy.io.votable` VOTable 文件读写

- `astropy.io.misc` 其他
- `astropy.io.registry` 注册
- `astropy.cosmology` 宇宙学
- `astropy.stats` 统计相关
- `astropy.config` 配置
- `astropy.utils` 工具

7.4 IPython

IPython [1] 是 Python 科学计算标准工具集的组成部分,它将所有的东西联系到了一起,它为交互式和探索式计算提供了一个强健而高效的环境。它是一个增强的 Python Shell。

IPython 给予用户一个非常友好的用户体验。在终端模式下,可以享受 tab 键带来的自动补全,而如果使用它的 notebook 模式时,它的用户体验更加优秀。

7.4.1 基础

IPython 的一些技巧:

- `TAB` 自动补全
- 在变量的前面或者后面加上一个问号(`?`)就可以显示该对象的一些通用信息

魔术命令

命令	说明
`%quickref`	显示 ipython 的快速参考
`%magic`	显示所有魔术命令的详细文档
`%debug`	Debug
`%hist`	命令历史
`%pdb`	在异常后自动进入调试器
`%paste`	执行剪切板中的 Python 代码
`%cpaste`	打开一个特殊提示符以便手工粘贴待执行的 Python 代码
`%reset`	删除命名空间所有变量/名称
`%run`	执行脚本
`%time`	执行一段代码的时间
`%who, %who_is, %whos`	显示命名空间定义的变量
`%xdel var`	删除 var 变量

matplotlab 集成

```
$ ipython --pylab
Python 2.7.6 (default, Nov 14 2013, 17:13:40)
```

[1] http://ipython.org/

```
Type "copyright", "credits" or "license" for more information.

IPython 1.1.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
Using matplotlib backend: MacOSX
```

记录输入与输出

IPython 可以记录整个控制台会话。执行 `%logstart` 即可开始记录日志。

```
In [1]: %logstart
Activating auto-logging. Current session state plus future input saved.
Filename       : ipython_log.py
Mode           : rotate
Output logging : False
Raw input log  : False
Timestamping   : False
State          : active
```

7.4.2 与操作系统交互

IPython 可以和操作系统的 Shell 紧密结合,在 IPython Shell 可以运行 Shell 命令或者脚本。

!cmd

```
In [2]: !ls -l pdr.fits
-rw-r--r--  1 hebl  staff  168626880 12  8 10:50 pdr.fits
```

相关的魔术命令

命令	说明
`!cmd`	执行 `cmd` 命令
`output = !cmd args`	执行 `cmd` ,并将 `stdout` 存入“output“中
`%alias aname cmd`	为 `cmd` 设置别名 `aname`
`%bookmark`	目录书签
`%cd directory`	更改工作目录
`%pwd`	返回当前工作目录
`%dhist`	打印目录访问历史
`%env`	返回系统环境变量,以 `dict` 格式显示

7.4.3 IPython HTML Notebook

2011 年,IPython 开发团队开发了一种基于 Web 技术的交互式计算环境,这就是 IPython Notebook,目前它已经成为一个相当棒的交互式计算工具,同时也是科研和教学的一种理想媒介。

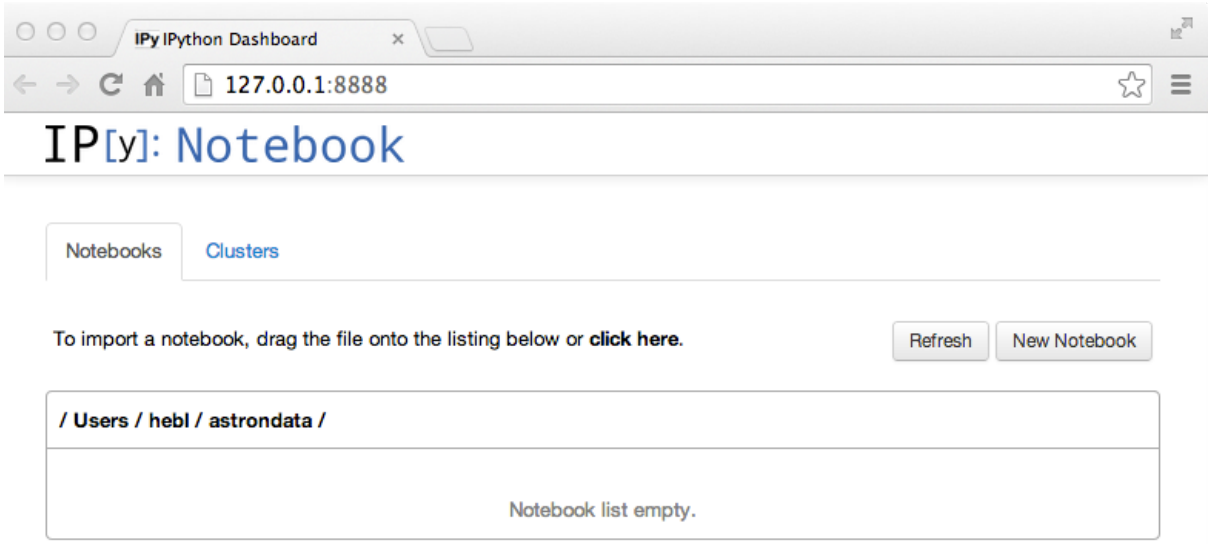
IPython Notebook 有一个基于 JSON 格式的 `.ipynb` 文档,可以轻松的分享代码、输出结果和图片。

IPython Notebook 甚至可以配置成一个 Notebook 服务器,用户可以通过 Web 界面登录 Notebook 平台进行交互式计算,或者分享 `.ipynb` 文档。

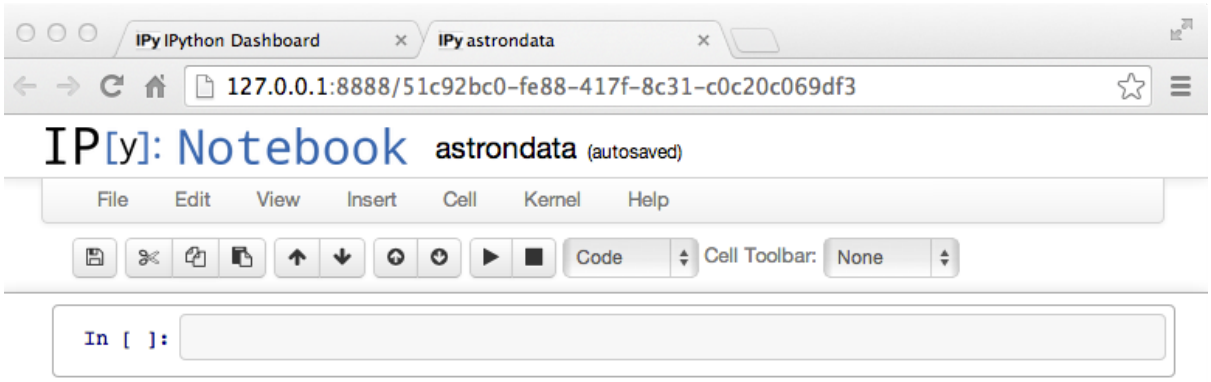
启动一个 IPython Notebook, 加上 `--pylab=inline` 可以将在绘图的结果直接展示在 notebook 上:

```
$ ipython notebook --pylab=inline
[NotebookApp] Using existing profile dir: u'/Users/hebl/.ipython/profile_default'
[NotebookApp] Using MathJax from CDN: http://cdn.mathjax.org/mathjax/latest/MathJax.j
[NotebookApp] Serving notebooks from local directory: /Users/hebl/astrondata
[NotebookApp] The IPython Notebook is running at: http://127.0.0.1:8888/
[NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to s
```

系统会自动打开一个页面



新建一个 notebook,并命名为 *astrondata*



IPython Notebook 快捷键

快捷键	说明
`Shift-Enter`	run cell
`Ctrl-Enter`	run cell in-place
`Alt-Enter`	run cell, insert below
`Ctrl-m x`	cut cell
`Ctrl-m c`	copy cell
`Ctrl-m v`	paste cell
`Ctrl-m d`	delete cell
`Ctrl-m z`	undo last cell deletion
`Ctrl-m -`	split cell
`Ctrl-m a`	insert cell above
`Ctrl-m b`	insert cell below
`Ctrl-m o`	toggle output
`Ctrl-m O`	toggle output scroll
`Ctrl-m l`	toggle line numbers
`Ctrl-m s`	save notebook
`Ctrl-m j`	move cell down
`Ctrl-m k`	move cell up
`Ctrl-m y`	code cell
`Ctrl-m m`	markdown cell
`Ctrl-m t`	raw cell
`Ctrl-m 1-6`	heading 1-6 cell
`Ctrl-m p`	select previous
`Ctrl-m n`	select next
`Ctrl-m i`	interrupt kernel
`Ctrl-m .`	restart kernel
`Ctrl-m h`	show keyboard shortcuts

使用

读取 PDR 星表文件

```
In [1]: import numpy as np
        import pandas as pd

In [2]: pdr = pd.read_csv('pdr.csv')

In [3]: pdr.head()

Out[3]: <class 'pandas.core.frame.DataFrame'>
        Int64Index: 5 entries, 0 to 4
        Data columns (total 34 columns):
        specid          5  non-null values
        designation     5  non-null values
        obsdate         5  non-null values
        lmjd            5  non-null values
        plate           5  non-null values
        spid            5  non-null values
        fiberid         5  non-null values
        ra              5  non-null values
        dec             5  non-null values
        snru            5  non-null values
        snrg            5  non-null values
        snrr            5  non-null values
        snri            5  non-null values
        snrz            5  non-null values
        objtype         5  non-null values
        class           5  non-null values
        subclass        5  non-null values
        magtype         5  non-null values
        mag1            5  non-null values
        mag2            5  non-null values
        mag3            5  non-null values
        mag4            5  non-null values
        mag5            5  non-null values
        mag6            5  non-null values
        mag7            5  non-null values
        tsource         5  non-null values
        fibertype       5  non-null values
        tfrom           5  non-null values
        tinfo           0  non-null values
        z               5  non-null values
        rv              5  non-null values
        z_err           0  non-null values
        elodierv        5  non-null values
        elodierv_err    5  non-null values
        dtypes: float64(20), int64(4), object(10)
```

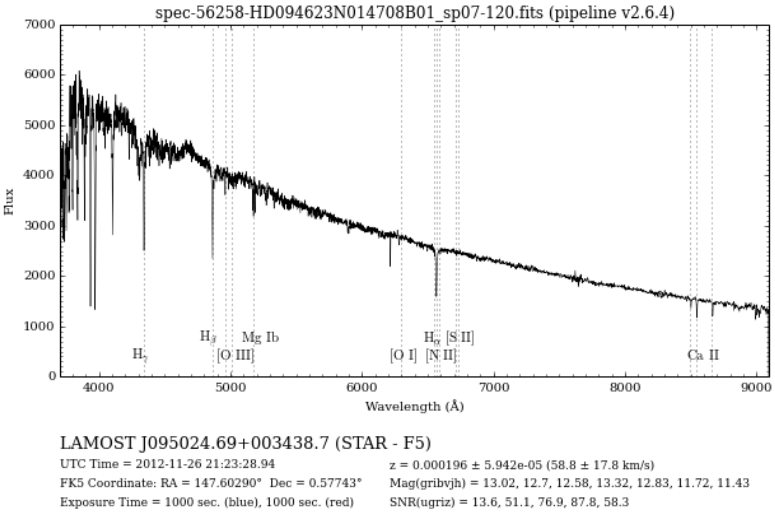
可以使用 `pdr.designation` 这样的格式读取一列数据,这个是 DataFrame 面向列的数据结构的一大亮点。

```
In [4]: pdr.designation
```

```
Out[4]: 0       J220716.38+000441.5
        1       J220545.34-012035.0
        2       J220800.83-003953.3
        3       J220832.29+002951.0
        4       J221403.28-005910.0
        5       J221652.14+005312.8
        6       J221552.14-002117.4
        7       J220142.30-002458.6
        8       J220652.19-010929.6
        9       J220013.50+011220.2
        10      J220709.56-012743.4
        11      J221712.60-002502.8
        12      J220503.88+000341.2
        13      J221646.47-003917.8
        14      J221008.08-005947.2
        ...
        717481    J154002.81+533151.6
        717482    J155453.81+521447.7
        717483    J155831.11+515744.1
        717484    J153732.75+500623.1
        717485    J153350.85+515533.0
        717486    J155421.92+533647.7
        717487    J153004.81+511602.8
        717488    J154723.04+501823.7
        717489    J155248.64+535918.7
        717490    J155356.46+522650.6
        717491    J154929.73+524542.2
        717492    J154036.77+533213.1
        717493    J154540.73+504021.0
        717494    J153012.70+520414.9
        717495    J155250.78+523536.9
        Name: designation, Length: 717496, dtype: object
```

绘制一张 LAMOST 的光谱图,程序由王靓博士 [2] 贡献。

```
In [4]: plot_file('spec-56258-HD094623N014708B01_sp07-120.fits')
```



[2] 王靓，wang.leon@gmail.com ，本段代码见：http://code.china-vo.org/training/astrodb-training2013/tree/master/python/plot_lamost.py

CHAPTER 8

PostgreSQL

PostgreSQL [1] 是一个知名的开源数据库系统,可以运行在几乎所有的开放平台上,并拥有与企业级数据库相媲美的特性,如完善的 SQL 标准支持、多版本并发控制、时间点恢复、表空间、异步复制、热备等。

8.1 数据类型

数据类型	别名	说明
smallint	int2	带符号短整形 (2 字节)
integer	int, int4	带符号整形 (4 字节)
bigint	int8	带符号长整形 (8 字节)
smallserial	serial2	自增短整形 (2 字节)
serial	serial4	自增整形 (4 字节)
bigserial	serial8	自增的长整形 (8 字节)
real	float4	单精度浮点数 (4 字节)
double precision	float8	双精度浮点数 (8 字节)
numeric [(p,s)]	decimal[(p,s)]	自定义精度数值
boolean	bool	逻辑类型 (true/false)
character varying [(n)]	varchar [(n)]	变长字符串
character [(n)]	char [(n)]	定长字符串
text		变长字符串
time [(p)] [without time zone]		一天中的某一时刻,无时区
time [(p)] with time zone	timetz	一天中的某一时刻,包含时区
timestamp [(p)] [without time zone]		日期时间,无时区
timestamp [(p)] with time zone	timestamptz	日期时间,包含时区

Continued on next page

[1] http://www.postgresql.org/

Table 8.1 – continued from previous page

数据类型	别名	说明
date		日期 (year, month, day)
interval [fields] [(p)]		时间段
bit [(n)]		定长二元数据类型
bit varying [(n)]	varbit	变长二元数据类型
bytea		二进制数据
inet		IPv4 或 IPv6 网络地址及掩码
cidr		IPv4 或者 IPv6 网络地址
macaddr		MAC 地址
money		固定精度货币
path		一系列点
point		点坐标
polygon		闭合多边形
box		矩形
line		无限长直线
lseg		线段
circle		圆
tsquery		文本搜索
tsvector		文档搜索
txid_snapshot		user-level transaction ID snapshot
uuid		通用统一标识符
xml		XML
json		JSON

NUMERIC(p,s) p 是长度,s 是精度,比如赤经 RA 一般的精度是长度 12,小数位 8,即是 NUMERIC(12,8) 。

8.2 基本命令

8.2.1 initdb

initdb 为一个新的 POstgreSQL 系统建立一个数据库目录结构。用法:

```
initdb -D path [options]
```

path 是数据库目录,必须确保 PostgreSQL 的运行用户,一般是 postgres 有完全权限,因此在执行上述命令前需要:

```
$ chown postgres:postgres path
```

initdb 会生成一个默认的 template1 数据库, 每次创建新的数据库时默认以 temnplate1 为模板。

8.2.2 psql

psql 是基本的一个命令行访问数据库的工具,而且是一个交互式前端工具,它的使用在后面的会频繁提及。

```
psql is the PostgreSQL interactive terminal.

Usage:
  psql [OPTION]... [DBNAME [USERNAME]]

General options:
  -c, --command=COMMAND    run only single command (SQL or internal) and exit
  -d, --dbname=DBNAME      database name to connect to (default: "hebl")
  -f, --file=FILENAME      execute commands from file, then exit
  -l, --list               list available databases, then exit
  -v, --set=, --variable=NAME=VALUE
                           set psql variable NAME to VALUE
  -V, --version            output version information, then exit
  -X, --no-psqlrc          do not read startup file (~/.psqlrc)
  -1 ("one"), --single-transaction
                           execute as a single transaction (if non-interactive)
  -?, --help               show this help, then exit

Input and output options:
  -a, --echo-all           echo all input from script
  -e, --echo-queries       echo commands sent to server
  -E, --echo-hidden        display queries that internal commands generate
  -L, --log-file=FILENAME  send session log to file
  -n, --no-readline        disable enhanced command line editing (readline)
  -o, --output=FILENAME    send query results to file (or |pipe)
  -q, --quiet              run quietly (no messages, only query output)
  -s, --single-step        single-step mode (confirm each query)
  -S, --single-line        single-line mode (end of line terminates SQL command)

Output format options:
  -A, --no-align           unaligned table output mode
  -F, --field-separator=STRING
                           set field separator (default: "|")
  -H, --html               HTML table output mode
  -P, --pset=VAR[=ARG]     set printing option VAR to ARG (see \pset command)
  -R, --record-separator=STRING
                           set record separator (default: newline)
  -t, --tuples-only        print rows only
```

```
  -T, --table-attr=TEXT    set HTML table tag attributes (e.g., width, border)
  -x, --expanded           turn on expanded table output
  -z, --field-separator-zero
                           set field separator to zero byte
  -0, --record-separator-zero
                           set record separator to zero byte

Connection options:
  -h, --host=HOSTNAME      database server host or socket directory (default: "local
  -p, --port=PORT          database server port (default: "5432")
  -U, --username=USERNAME  database user name (default: "hebl")
  -w, --no-password        never prompt for password
  -W, --password           force password prompt (should happen automatically)

For more information, type "\?" (for internal commands) or "\help" (for SQL
commands) from within psql, or consult the psql section in the PostgreSQL
documentation.
```

常用的使用方法,以 `user` 用户连接数据库 `dbname`

```
psql -U user dbname
```

不加参数 `-U` 的话,默认用户是当前使用 `shell` 的用户。

常用的选项参数:

选项	说明
`-U` *user*	以指定用户连接数据库
`-h` *hostname*	连接到指定主机,可以是域名,也可以是 IP 地址
`-p` *port*	使用指定端口连接
`-d` *dbname*	指定连接的数据库,未指定数据库,默认数据库名位 *user*
`-c` *command*	指定 psql 执行 `command` 指令
`-f` *filename*	读取指定文件,并执行其中的 SQL 语句,等同于 `< filename`
`-o` *filename*	输出到指定文件,等同于 `> filename`
`-H`	以 HTML 格式输出查询结果
`-A`	输出格式布局非对齐,在输出成 csv 文件时有用
`-F` *sep*	输出的字段分隔符,默认 `'|'` ,csv 格式的可以指定为 `','`
`-t`	仅返回数据,不输出列名,和结果信息。
`-x`	扩展行模式

psql shell,连接上数据库以后,就进入 `psql shell` 模式,在这个模式中,大部分的命令前面都有反斜杠 `\`

```
General
  \copyright             show PostgreSQL usage and distribution terms
  \g [FILE] or ;         execute query (and send results to file or |pipe)
  \gset [PREFIX]         execute query and store results in psql variables
```

```
  \h [NAME]              help on syntax of SQL commands, * for all commands
  \q                     quit psql
  \watch [SEC]           execute query every SEC seconds

Query Buffer
  \e [FILE] [LINE]       edit the query buffer (or file) with external editor
  \ef [FUNCNAME [LINE]]  edit function definition with external editor
  \p                     show the contents of the query buffer
  \r                     reset (clear) the query buffer
  \s [FILE]              display history or save it to file
  \w FILE                write query buffer to file

Input/Output
  \copy ...              perform SQL COPY with data stream to the client host
  \echo [STRING]         write string to standard output
  \i FILE                execute commands from file
  \ir FILE               as \i, but relative to location of current script
  \o [FILE]              send all query results to file or |pipe
  \qecho [STRING]        write string to query output stream (see \o)

Informational
  (options: S = show system objects, + = additional detail)
  \d[S+]                 list tables, views, and sequences
  \d[S+]  NAME           describe table, view, sequence, or index
  \da[S]  [PATTERN]      list aggregates
  \db[+]  [PATTERN]      list tablespaces
  \dc[S+] [PATTERN]      list conversions
  \dC[+]  [PATTERN]      list casts
  \dd[S]  [PATTERN]      show object descriptions not displayed elsewhere
  \ddp    [PATTERN]      list default privileges
  \dD[S+] [PATTERN]      list domains
  \det[+] [PATTERN]      list foreign tables
  \des[+] [PATTERN]      list foreign servers
  \deu[+] [PATTERN]      list user mappings
  \dew[+] [PATTERN]      list foreign-data wrappers
  \df[antw][S+] [PATRN]  list [only agg/normal/trigger/window] functions
  \dF[+]  [PATTERN]      list text search configurations
  \dFd[+] [PATTERN]      list text search dictionaries
  \dFp[+] [PATTERN]      list text search parsers
  \dFt[+] [PATTERN]      list text search templates
  \dg[+]  [PATTERN]      list roles
  \di[S+] [PATTERN]      list indexes
  \dl                    list large objects, same as \lo_list
```

```
  \dL[S+] [PATTERN]      list procedural languages
  \dm[S+] [PATTERN]      list materialized views
  \dn[S+] [PATTERN]      list schemas
  \do[S]  [PATTERN]      list operators
  \dO[S+] [PATTERN]      list collations
  \dp     [PATTERN]      list table, view, and sequence access privileges
  \drds [PATRN1 [PATRN2]] list per-database role settings
  \ds[S+] [PATTERN]      list sequences
  \dt[S+] [PATTERN]      list tables
  \dT[S+] [PATTERN]      list data types
  \du[+]  [PATTERN]      list roles
  \dv[S+] [PATTERN]      list views
  \dE[S+] [PATTERN]      list foreign tables
  \dx[+]  [PATTERN]      list extensions
  \dy     [PATTERN]      list event triggers
  \l[+]   [PATTERN]      list databases
  \sf[+] FUNCNAME        show a function's definition
  \z      [PATTERN]      same as \dp

Formatting
  \a                     toggle between unaligned and aligned output mode
  \C [STRING]            set table title, or unset if none
  \f [STRING]            show or set field separator for unaligned query output
  \H                     toggle HTML output mode (currently off)
  \pset NAME [VALUE]     set table output option
                         (NAME := {format|border|expanded|fieldsep|fieldsep_zero|foot
                         numericlocale|recordsep|recordsep_zero|tuples_only|title|tab
  \t [on|off]            show only rows (currently off)
  \T [STRING]            set HTML <table> tag attributes, or unset if none
  \x [on|off|auto]       toggle expanded output (currently off)

Connection
  \c[onnect] [DBNAME|- USER|- HOST|- PORT|-]
                         connect to new database (currently "dr1")
  \encoding [ENCODING]   show or set client encoding
  \password [USERNAME]   securely change the password for a user
  \conninfo              display information about current connection

Operating System
  \cd [DIR]              change the current working directory
  \setenv NAME [VALUE]   set or unset environment variable
  \timing [on|off]       toggle timing of commands (currently off)
  \! [COMMAND]           execute command in shell or start interactive shell
```

```
Variables
  \prompt [TEXT] NAME    prompt user to set internal variable
  \set [NAME [VALUE]]    set internal variable, or list all if no parameters
  \unset NAME            unset (delete) internal variable

Large Objects
  \lo_export LOBOID FILE
  \lo_import FILE [COMMENT]
  \lo_list
  \lo_unlink LOBOID      large object operations
```

常用的用法包括:

- \l[+] 显示所有的数据库情况;
- \dn[+] 显示当前数据库的 schema 列表;
- \dt[+] schema. 显示 schema 下数据表的列表,无 schema 情况下,默认 public 模式;
- \d[+] schema.table 显示表 schema.table 的表结构;
- \dv[+] schema.view 显示视图 schema.view 的视图结构;

在这里 [+] 会显示更多的信息,比如空间占用量等。

8.2.3 pg_dump

pg_dump 可以将数据库的数据,结构、索引等信息导出到一个文件中,默认导出到 stdout ,可以通过管道符号轻易的转向到一个文件。

```
pg_dump dumps a database as a text file or to other formats.

Usage:
  pg_dump [OPTION]... [DBNAME]

General options:
  -f, --file=FILENAME          output file or directory name
  -F, --format=c|d|t|p         output file format (custom, directory, tar,
                               plain text (default))
  -j, --jobs=NUM               use this many parallel jobs to dump
  -v, --verbose                verbose mode
  -V, --version                output version information, then exit
  -Z, --compress=0-9           compression level for compressed formats
  --lock-wait-timeout=TIMEOUT  fail after waiting TIMEOUT for a table lock
  -?, --help                   show this help, then exit

Options controlling the output content:
```

```
  -a, --data-only              dump only the data, not the schema
  -b, --blobs                  include large objects in dump
  -c, --clean                  clean (drop) database objects before recreating
  -C, --create                 include commands to create database in dump
  -E, --encoding=ENCODING      dump the data in encoding ENCODING
  -n, --schema=SCHEMA          dump the named schema(s) only
  -N, --exclude-schema=SCHEMA  do NOT dump the named schema(s)
  -o, --oids                   include OIDs in dump
  -O, --no-owner               skip restoration of object ownership in
                               plain-text format
  -s, --schema-only            dump only the schema, no data
  -S, --superuser=NAME         superuser user name to use in plain-text format
  -t, --table=TABLE            dump the named table(s) only
  -T, --exclude-table=TABLE    do NOT dump the named table(s)
  -x, --no-privileges          do not dump privileges (grant/revoke)
  --binary-upgrade             for use by upgrade utilities only
  --column-inserts             dump data as INSERT commands with column names
  --disable-dollar-quoting     disable dollar quoting, use SQL standard quoting
  --disable-triggers           disable triggers during data-only restore
  --exclude-table-data=TABLE   do NOT dump data for the named table(s)
  --inserts                    dump data as INSERT commands, rather than COPY
  --no-security-labels         do not dump security label assignments
  --no-synchronized-snapshots  do not use synchronized snapshots in parallel jobs
  --no-tablespaces             do not dump tablespace assignments
  --no-unlogged-table-data     do not dump unlogged table data
  --quote-all-identifiers      quote all identifiers, even if not key words
  --section=SECTION            dump named section (pre-data, data, or post-data)
  --serializable-deferrable    wait until the dump can run without anomalies
  --use-set-session-authorization
                               use SET SESSION AUTHORIZATION commands instead of
                               ALTER OWNER commands to set ownership

Connection options:
  -d, --dbname=DBNAME      database to dump
  -h, --host=HOSTNAME      database server host or socket directory
  -p, --port=PORT          database server port number
  -U, --username=NAME      connect as specified database user
  -w, --no-password        never prompt for password
  -W, --password           force password prompt (should happen automatically)
  --role=ROLENAME          do SET ROLE before dump
```

一些基本的连接参数与 `psql` 相同。

选项	说明
-a	只导出数据,不导出结构
-n *schema*	只导出模式是 schema 的数据表
-O	忽略不导出数据的所有权信息
-s	只导出数据库结构,不导出数据
-t *table*	只导出数据表 table
-T *table*	不导出 table
-x	不导出权限信息

注

多个模式、表的话,可以多次使用一个选项,比如:

```
-t table1 -t table2 -n schema1 -n schema2
```

导出并打包成 gz

```
pg_dump -U user database -n schema | gzip > out.sql.gz
```

导出的数据库,可以被重新导入,这在数据库升级或者,数据库复制时有用。

```
pg_restore -U user database < out.sql
```

如果需要导出所有的数据库的话,可以使用 pg_dumpall 命令。

8.3 SQL 指令

SQL 是数据库的操作语言。

8.3.1 数据库 DATABASE

创建数据库:

```
Command:     CREATE DATABASE
Description: create a new database
Syntax:
CREATE DATABASE name
    [ [ WITH ] [ OWNER [=] user_name ]
           [ TEMPLATE [=] template ]
           [ ENCODING [=] encoding ]
           [ LC_COLLATE [=] lc_collate ]
           [ LC_CTYPE [=] lc_ctype ]
           [ TABLESPACE [=] tablespace_name ]
           [ CONNECTION LIMIT [=] connlimit ] ]
```

修改数据库:

```
Command:     ALTER DATABASE
Description: change a database
Syntax:
ALTER DATABASE name [ [ WITH ] option [ ... ] ]

where option can be:

    CONNECTION LIMIT connlimit

ALTER DATABASE name RENAME TO new_name

ALTER DATABASE name OWNER TO new_owner

ALTER DATABASE name SET TABLESPACE new_tablespace

ALTER DATABASE name SET configuration_parameter { TO | = } { value | DEFAULT }
ALTER DATABASE name SET configuration_parameter FROM CURRENT
ALTER DATABASE name RESET configuration_parameter
ALTER DATABASE name RESET ALL
```

8.3.2 模式 SCHEMA

数据库下可以有多个模式,模式可以理解为一种形式的目录,SQL 对象(表、视图、索引、序列等)都包含在模式中。这是现代数据库系统提供的一个关系命名机制。

有了模式,就可以唯一的标识一个关系(表、视图、索引、序列等):

```
schema.table
schema.view
```

PostgreSQL 自动产生并且默认的一个 `schema` 是 `public` ,因此:

```
public.catalogue  等同于 catalogue
```

创建模式:

```
Command:     CREATE SCHEMA
Description: define a new schema
Syntax:
CREATE SCHEMA schema_name [ AUTHORIZATION user_name ] [ schema_element [ ... ] ]
CREATE SCHEMA AUTHORIZATION user_name [ schema_element [ ... ] ]
CREATE SCHEMA IF NOT EXISTS schema_name [ AUTHORIZATION user_name ]
CREATE SCHEMA IF NOT EXISTS AUTHORIZATION user_name
```

修改模式:

```
Command:     ALTER SCHEMA
```

```
Description: change the definition of a schema
Syntax:
ALTER SCHEMA name RENAME TO new_name
ALTER SCHEMA name OWNER TO new_owner
```

8.3.3 表 TABLE

表是数据库存储的基本单元,表拥有表结构,结构由一系列的字段组成。

建表:

```
Command:     CREATE TABLE
Description: define a new table
Syntax:
CREATE [ [ GLOBAL | LOCAL ] { TEMPORARY | TEMP } | UNLOGGED ] TABLE [ IF NOT EXISTS ]
  { column_name data_type [ COLLATE collation ] [ column_constraint [ ... ] ]
    | table_constraint
    | LIKE source_table [ like_option ... ] }
    [, ... ]
] )
[ INHERITS ( parent_table [, ... ] ) ]
[ WITH ( storage_parameter [= value] [, ... ] ) | WITH OIDS | WITHOUT OIDS ]
[ ON COMMIT { PRESERVE ROWS | DELETE ROWS | DROP } ]
[ TABLESPACE tablespace_name ]

CREATE [ [ GLOBAL | LOCAL ] { TEMPORARY | TEMP } | UNLOGGED ] TABLE [ IF NOT EXISTS ]
    OF type_name [ (
  { column_name WITH OPTIONS [ column_constraint [ ... ] ]
    | table_constraint }
    [, ... ]
) ]
[ WITH ( storage_parameter [= value] [, ... ] ) | WITH OIDS | WITHOUT OIDS ]
[ ON COMMIT { PRESERVE ROWS | DELETE ROWS | DROP } ]
[ TABLESPACE tablespace_name ]

where column_constraint is:

[ CONSTRAINT constraint_name ]
{ NOT NULL |
  NULL |
  CHECK ( expression ) [ NO INHERIT ] |
  DEFAULT default_expr |
  UNIQUE index_parameters |
  PRIMARY KEY index_parameters |
```

```
  REFERENCES reftable [ ( refcolumn ) ] [ MATCH FULL | MATCH PARTIAL | MATCH SIMPLE ]
    [ ON DELETE action ] [ ON UPDATE action ] }
[ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]

and table_constraint is:

[ CONSTRAINT constraint_name ]
{ CHECK ( expression ) [ NO INHERIT ] |
  UNIQUE ( column_name [, ... ] ) index_parameters |
  PRIMARY KEY ( column_name [, ... ] ) index_parameters |
  EXCLUDE [ USING index_method ] ( exclude_element WITH operator [, ... ] ) index_par
  FOREIGN KEY ( column_name [, ... ] ) REFERENCES reftable [ ( refcolumn [, ... ] ) ]
    [ MATCH FULL | MATCH PARTIAL | MATCH SIMPLE ] [ ON DELETE action ] [ ON UPDATE ac
[ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]

and like_option is:

{ INCLUDING | EXCLUDING } { DEFAULTS | CONSTRAINTS | INDEXES | STORAGE | COMMENTS | A

index_parameters in UNIQUE, PRIMARY KEY, and EXCLUDE constraints are:

[ WITH ( storage_parameter [= value] [, ... ] ) ]
[ USING INDEX TABLESPACE tablespace_name ]

exclude_element in an EXCLUDE constraint is:

{ column_name | ( expression ) } [ opclass ] [ ASC | DESC ] [ NULLS { FIRST | LAST } ]
```

修改表:

```
Command:     ALTER TABLE
Description: change the definition of a table
Syntax:
ALTER TABLE [ IF EXISTS ] [ ONLY ] name [ * ]
    action [, ... ]
ALTER TABLE [ IF EXISTS ] [ ONLY ] name [ * ]
    RENAME [ COLUMN ] column_name TO new_column_name
ALTER TABLE [ IF EXISTS ] [ ONLY ] name [ * ]
    RENAME CONSTRAINT constraint_name TO new_constraint_name
ALTER TABLE [ IF EXISTS ] name
    RENAME TO new_name
ALTER TABLE [ IF EXISTS ] name
    SET SCHEMA new_schema
```

```
where action is one of:

    ADD [ COLUMN ] column_name data_type [ COLLATE collation ] [ column_constraint [
    DROP [ COLUMN ] [ IF EXISTS ] column_name [ RESTRICT | CASCADE ]
    ALTER [ COLUMN ] column_name [ SET DATA ] TYPE data_type [ COLLATE collation ] [
    ALTER [ COLUMN ] column_name SET DEFAULT expression
    ALTER [ COLUMN ] column_name DROP DEFAULT
    ALTER [ COLUMN ] column_name { SET | DROP } NOT NULL
    ALTER [ COLUMN ] column_name SET STATISTICS integer
    ALTER [ COLUMN ] column_name SET ( attribute_option = value [, ... ] )
    ALTER [ COLUMN ] column_name RESET ( attribute_option [, ... ] )
    ALTER [ COLUMN ] column_name SET STORAGE { PLAIN | EXTERNAL | EXTENDED | MAIN }
    ADD table_constraint [ NOT VALID ]
    ADD table_constraint_using_index
    VALIDATE CONSTRAINT constraint_name
    DROP CONSTRAINT [ IF EXISTS ]  constraint_name [ RESTRICT | CASCADE ]
    DISABLE TRIGGER [ trigger_name | ALL | USER ]
    ENABLE TRIGGER [ trigger_name | ALL | USER ]
    ENABLE REPLICA TRIGGER trigger_name
    ENABLE ALWAYS TRIGGER trigger_name
    DISABLE RULE rewrite_rule_name
    ENABLE RULE rewrite_rule_name
    ENABLE REPLICA RULE rewrite_rule_name
    ENABLE ALWAYS RULE rewrite_rule_name
    CLUSTER ON index_name
    SET WITHOUT CLUSTER
    SET WITH OIDS
    SET WITHOUT OIDS
    SET ( storage_parameter = value [, ... ] )
    RESET ( storage_parameter [, ... ] )
    INHERIT parent_table
    NO INHERIT parent_table
    OF type_name
    NOT OF
    OWNER TO new_owner
    SET TABLESPACE new_tablespace

and table_constraint_using_index is:

    [ CONSTRAINT constraint_name ]
    { UNIQUE | PRIMARY KEY } USING INDEX index_name
    [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
```

建表的时候可以定义一些约束条件,比如主键(PRIMARY KEY),外键(FOREIGN KEY),唯一键(UNIQUE),约束(CHECK)等。但是在建立记录固定的天文数据库时,建议先建立数据表结构,然后导入数据,最后再加约束和索引。

8.3.4 视图 VIEW

建立视图:

```
Command:     CREATE VIEW
Description: define a new view
Syntax:
CREATE [ OR REPLACE ] [ TEMP | TEMPORARY ] [ RECURSIVE ] VIEW name [ ( column_name [,
    [ WITH ( view_option_name [= view_option_value] [, ... ] ) ]
    AS query
```

一个例子,根据 LAMOST PDR 的数据表结构,构建一个 view,可以查询光谱文件的全名:

```
CREATE VIEW specFits AS
        SELECT designation, planId, lmjd, spId,fiberId,
        (planId::text || '/spec-'::text || lmjd || '-'::text || planId::text
        || '_sp'::text
        || to_char(spId::double precision, 'FM00'::text) || '-'::text
        || to_char(fiberId::double precision, 'FM000'::text)
        || '.fits'::text) AS file

        FROM catalogue;
```

查看 VIEW 的结构:

```
sql> \d+ specFits
                          View "public.specfits"
   Column    |         Type          | Modifiers | Storage  | Description
-------------+-----------------------+-----------+----------+-------------
 designation | character varying(30) |           | extended |
 planid      | character varying(30) |           | extended |
 lmjd        | integer               |           | plain    |
 spid        | smallint              |           | plain    |
 fiberid     | smallint              |           | plain    |
 file        | text                  |           | extended |
View definition:
 SELECT catalogue.designation,
    catalogue.planid,
    catalogue.lmjd,
    catalogue.spid,
    catalogue.fiberid,
```

```
    (((((((catalogue.planid::text || '/spec-'::text) || catalogue.lmjd)
        || '-'::text)
        || catalogue.planid::text)
        || '_sp'::text)
        || to_char(catalogue.spid::double precision, 'FM00'::text))
        || '-'::text)
        || to_char(catalogue.fiberid::double precision, 'FM000'::text))
        || '.fits'::text AS file
   FROM catalogue;
sql> SELECT designation, file FROM specFits LIMIT 10;
     designation     |                    file
---------------------+-------------------------------------------
 J031553.29+522639.2 | B87806_1/spec-55878-B87806_1_sp09-013.fits
 J032459.64+511215.0 | B87806_1/spec-55878-B87806_1_sp06-120.fits
 J031711.05+510142.3 | B87806_1/spec-55878-B87806_1_sp08-108.fits
 J030953.95+530028.8 | B87806_1/spec-55878-B87806_1_sp15-197.fits
 J031445.85+531551.4 | B87806_1/spec-55878-B87806_1_sp09-121.fits
 J030740.45+534510.6 | B87806_1/spec-55878-B87806_1_sp16-122.fits
 J030450.51+512832.3 | B87806_1/spec-55878-B87806_1_sp03-245.fits
 J031444.68+502315.6 | B87806_1/spec-55878-B87806_1_sp01-103.fits
 J032154.18+514025.2 | B87806_1/spec-55878-B87806_1_sp06-082.fits
 J031031.82+542218.6 | B87806_1/spec-55878-B87806_1_sp11-144.fits
(10 rows)
```

视图是虚表,它可以让我们构造一些非常有用的查询结构,优化数据库的设计。

8.3.5 索引 INDEX

针对一个小的数据表,数据量数百行,有无索引对用户体验来说,无特别明显的区别,但是对于数万、乃至数亿的数据来说,索引就显得非常重要,它可以非常显著的提升检索数据的速度,这个用户可以尝试下。

普通字段的索引一般采用 `B-Tree` 索引结构,它基本可以理解为实线性的,但对于空间坐标来说,`(RA,Dec)` 是一个空间点,索引将是一个二维的,如果建立一个高效的索引非常重要,这里我们就使用 `pgSphere` 的 `GIST` 索引。

索引建立方法:

```
Command:     CREATE INDEX
Description: define a new index
Syntax:
CREATE [ UNIQUE ] INDEX [ CONCURRENTLY ] [ name ] ON table_name [ USING method ]
    ( { column_name | ( expression ) } [ COLLATE collation ] [ opclass ] [ ASC | DESC
    [ WITH ( storage_parameter = value [, ... ] ) ]
    [ TABLESPACE tablespace_name ]
```

```
    [ WHERE predicate ]
```

例如字段 lmjd :

```
CREATE index pdr_idx_lmjd ON catalogue (lmjd);
```

空间索引:

```
CREATE INDEX pdr_idx_spos ON catalogue USING GIST(spos(RA,Dec));
```

8.3.6 数据导入导出 COPY

数据录入,尤其是以 CSV 等格式的海量数据时,推荐使用 COPY 指令进行数据导入。

```
gawk 'NR>1 {print}' pdr.csv | psql pdr
        -c "COPY catalogue FROM STDIN USING DELIMITERS ',' NULL ''"

cat pdr.csv | psql pdr -c "COPY catalogue FROM STDIN WITH CSV HEADER"
```

我们先看看 COPY 指令是如何定义的:

```
Command:     COPY
Description: copy data between a file and a table
Syntax:
COPY table_name [ ( column_name [, ...] ) ]
    FROM { 'filename' | PROGRAM 'command' | STDIN }
    [ [ WITH ] ( option [, ...] ) ]

COPY { table_name [ ( column_name [, ...] ) ] | ( query ) }
    TO { 'filename' | PROGRAM 'command' | STDOUT }
    [ [ WITH ] ( option [, ...] ) ]

where option can be one of:

    FORMAT format_name
    OIDS [ boolean ]
    FREEZE [ boolean ]
    DELIMITER 'delimiter_character'
    NULL 'null_string'
    HEADER [ boolean ]
    QUOTE 'quote_character'
    ESCAPE 'escape_character'
    FORCE_QUOTE { ( column_name [, ...] ) | * }
    FORCE_NOT_NULL ( column_name [, ...] )
    ENCODING 'encoding_name'
```

由于数据库的权限问题,建议使用 COPY 做导入操作是使用 STDIN 和 cat 配合。

8.3.7 数据检索 SELECT

检索指令 SELECT 用法:

```
Command:     SELECT
Description: retrieve rows from a table or view
Syntax:
[ WITH [ RECURSIVE ] with_query [, ...] ]
SELECT [ ALL | DISTINCT [ ON ( expression [, ...] ) ] ]
    * | expression [ [ AS ] output_name ] [, ...]
    [ FROM from_item [, ...] ]
    [ WHERE condition ]
    [ GROUP BY expression [, ...] ]
    [ HAVING condition [, ...] ]
    [ WINDOW window_name AS ( window_definition ) [, ...] ]
    [ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
    [ ORDER BY expression [ ASC | DESC | USING operator ] [ NULLS { FIRST | LAST } ]
    [ LIMIT { count | ALL } ]
    [ OFFSET start [ ROW | ROWS ] ]
    [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ]
    [ FOR { UPDATE | NO KEY UPDATE | SHARE | KEY SHARE } [ OF table_name [, ...] ] [

where from_item can be one of:

    [ ONLY ] table_name [ * ] [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
    [ LATERAL ] ( select ) [ AS ] alias [ ( column_alias [, ...] ) ]
    with_query_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
    [ LATERAL ] function_name ( [ argument [, ...] ] ) [ AS ] alias [ ( column_alias
    [ LATERAL ] function_name ( [ argument [, ...] ] ) AS ( column_definition [, ...]
    from_item [ NATURAL ] join_type from_item [ ON join_condition | USING ( join_colu

and with_query is:

    with_query_name [ ( column_name [, ...] ) ] AS ( select | values | insert | updat

TABLE [ ONLY ] table_name [ * ]
```

8.3.8 数据更新 UPDATE

```
Command:     UPDATE
Description: update rows of a table
Syntax:
[ WITH [ RECURSIVE ] with_query [, ...] ]
```

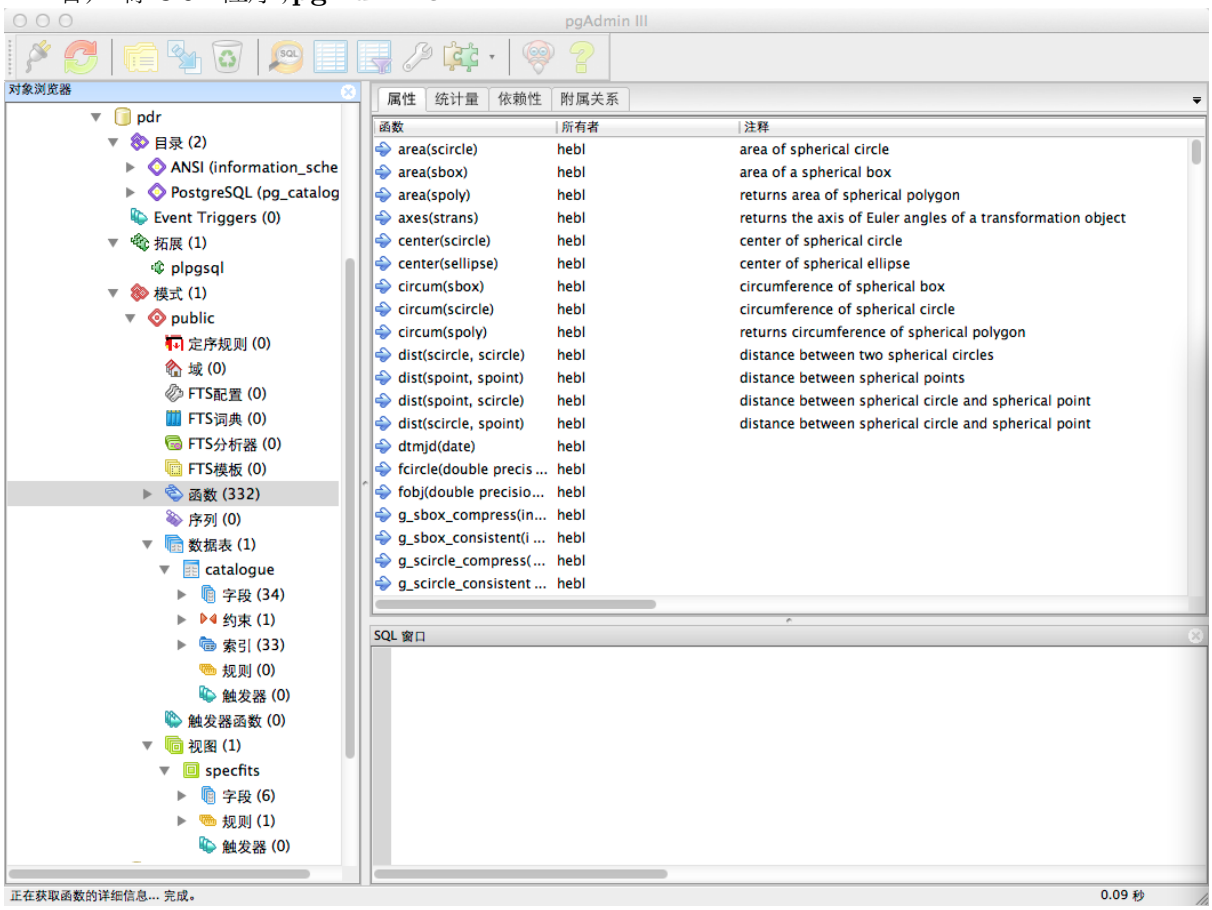
```
UPDATE [ ONLY ] table_name [ * ] [ [ AS ] alias ]
    SET { column_name = { expression | DEFAULT } |
          ( column_name [, ...] ) = ( { expression | DEFAULT } [, ...] ) } [, ...]
    [ FROM from_list ]
    [ WHERE condition | WHERE CURRENT OF cursor_name ]
    [ RETURNING * | output_expression [ [ AS ] output_name ] [, ...] ]
```

8.3.9 数据删除 DELETE

```
Command:     DELETE
Description: delete rows of a table
Syntax:
[ WITH [ RECURSIVE ] with_query [, ...] ]
DELETE FROM [ ONLY ] table_name [ * ] [ [ AS ] alias ]
    [ USING using_list ]
    [ WHERE condition | WHERE CURRENT OF cursor_name ]
    [ RETURNING * | output_expression [ [ AS ] output_name ] [, ...] ]
```

8.4 pgAdmin

客户端 GUI 程序,**pgAdmin3**



8.5 pgSphere

pgSphere [2] 是一个高效的 PostgreSQL 扩展库,它为 PostgreSQL 提供了球面数据类型和函数。安装 pgSphere 见第 1 章。pgSphere 是 PostgreSQL 的一个扩展,由 C 语言写成。

pgSphere 可以做:

- 球面数据输入输出
- 包含、覆盖以及其他操作
- 球面转换
- 球面数据类型索引

建立一个支持 pgSphere 的数据库

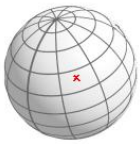
```
$ psql -U postgres database < pg_sphere.sql
```

由于版本的更新,请使用培训网站上的 `pg_sphere.sql` 文件。

8.5.1 数据类型

SQL type name	spherical type
spoint	point (position) 球面坐标点
strans	Euler transformation
scircle	circle 球面圆
sline	line 球面线段
sellipse	ellipse
spoly	polygon
spath	path
sbox	coordinate range 球面坐标范围

1. 球面坐标点 spoint



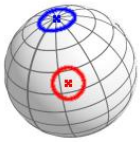
格式 spoint '(lng, lat)'

spoint 是最重要的一个数据类型,由经度、纬度组成,默认单位是弧度。经度范围 0-2π,纬度范围 -π/2 ~ π/2。使用的时候,也支持以 DMS 或者 HMS 格式。

```
sql> SELECT spoint '(0.1,-0.2)';
sql> SELECT spoint '( 10.1d, -90d)';
sql> SELECT spoint '( 10d 12m 11.3s, -13d 14m)';
sql> SELECT spoint '( 23h 44m 10s, -1.4321 )';
```

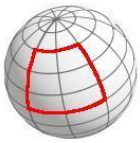
[2] http://pgsphere.projects.pgfoundry.org/

2. 球面圆 scircle



格式 `scircle '<(longitude, latitude), radius>'`

3. 球面坐标范围 sbox



格式 `sbox '(pos_sw, pos_ne)'`

8.5.2 构造函数

1. 构造球面点

`spoint(float8 lng, float8 lat)`

输入的单位是弧度,可以再自定义一个 SQL 函数来使用 deg

```
CREATE OR REPLACE FUNCTION spos(FLOAT8, FLOAT8) RETURNS spoint AS $$
    BEGIN
        RETURN spoint(radians($1), radians($2));
    END;
$$ LANGUAGE plpgsql IMMUTABLE;
```

使用:

```
sql> SELECT spos(180,0);
          spos
------------------------
 (3.14159265358979 , 0)
```

2. 构造球面圆

`scircle(spoint center, float8 radius)`

类似上面,我们也可以定义一个使用 deg 的函数:

```
CREATE OR REPLACE FUNCTION fCircle(FLOAT8, FLOAT8, FLOAT8) RETURNS scircle AS $$
        BEGIN
        RETURN scircle(spoint(radians($1), radians($2)), radians($3));
        END;
$$ LANGUAGE plpgsql IMMUTABLE;
```

8.5.3 运算符

在球面上免不了一些计算,pgSphere 引入了一些包含和覆盖的函数。

运算符	如果满足下面的条件,返回 true
@	运算符右边的对象包含左边的
~	运算符左边的对象包含右边的
!@	运算符右边的对象不包含左边的
!~	运算符左边的对象不包含右边的
&&	运算符左右的对象互相包含
!&&	运算符左右的对象互相不包含

应用案例 @ 在实现锥形检索时非常有用,比如:

```
sql> SELECT specId,planId,RA,Dec,class FROM catalogue
        WHERE spos(RA,Dec) @ scircle '<(332.00d,-0.664d), 0.2d>';
 specid | planid |      ra      |     dec     |  class
--------+--------+--------------+-------------+---------
 105040 | F5902  | 332.00348000 | -0.66481000 | Unknown
 105137 | F5902  | 332.14139000 | -0.70628000 | Unknown
 105050 | F5902  | 331.94492000 | -0.80145000 | Unknown
 105151 | F5902  | 331.95630000 | -0.47038000 | Unknown
 105028 | F5902  | 331.89249000 | -0.61879000 | GALAXY
 105156 | F5902  | 332.14185000 | -0.59118000 | Unknown
 105041 | F5902  | 331.84024000 | -0.63403000 | Unknown
 105163 | F5902  | 332.06512900 | -0.50408500 | STAR
 105047 | F5902  | 331.82643300 | -0.75629600 | STAR
 105168 | F5902  | 332.08339300 | -0.51841200 | STAR
 105027 | F5902  | 331.90488400 | -0.70474700 | STAR
(11 rows)
```

8.5.4 球面距离

计算球面距离使用的运算符是 <-> ,返回为弧度。可以是两个点的距离,也可以是球面圆,也可以是点和圆。如果点在圆内,则距离为 0。

```
sql> select (spoint '(332.14185d, -0.59118d)'
        <-> scircle '<(332.00d,-0.664d), 0.2d>');
 ?column?
----------
        0
(1 row)

sql> select (spoint '(332.14185d, -0.59118d)'
        <-> scircle '<(332.00d,0.664d), 0.2d>');
```

```
      ?column?
--------------------
 0.0185558100525635
(1 row)

sql> select degrees(spoint '(332.14185d, -0.59118d)'
        <-> scircle '<(332.00d,0.664d), 0.2d>') ;
     degrees
------------------
 1.06316960145832
(1 row)
```

8.5.5 其他函数

取经纬度:

```
long(spoint p)
lat(spoint p)
```

8.5.6 创建索引

pgSphere 使用 `GIST` 来创建索引,不同于普通的 `B-Tree` 。

```
CREATE INDEX pdr_idx_spos ON catalogue USING GIST(spos(RA,Dec))
```

创建索引可以大大提高 `@, &&, #, =, !=` 等运算符的执行速度。针对千万量级的数据表,pgSphere 表现相当不错,即使到数亿的量级,通过一些分区等方法,也可以实现很高的性能。

Part III

玩转数据

CHAPTER 9

文本文件

文本文件的操作,我们可以使用一些基本的 Unix/Linux 命令来进行,比如 cat 、awk 、sed 、wc 、head 、tail 等。首先介绍一些基础知识,如管道、文件输入输出等,接下来以 LAMOST PDR 星表 CSV 文件为例进行文本文件的操作,文件名是 pdr.csv 。

9.1 基础知识

9.1.1 IO 与重定向

Shell 环境下文件的数据输入输出。

< 是输入,> 是输出。

9.1.2 管道

管道 | ,顾名思义,就像一个管子,一端可以输入,另一端是输出,在 Shell 环境下,一个常用的场景是一个程序的输出是另外一个程序的输入,通过管道就可以轻松的连接起来。管道符号左边是输入,右边是输出。

一个例子

```
cat pdr.csv | wc
```

9.2 查看文件

查看完整文件:cat、more、less

```
cat pdr.csv
```

查看首行,默认为 10 行:head

```
head -n 10 pdr.csv
```

查看尾行,默认为 10 行:tail

```
tail -n 10 pdr.csv
```

9.3 筛选文件

筛选文件常用的指令是 awk,awk 的基本用法

```
Usage: gawk [POSIX or GNU style options] -f progfile [--] file ...
Usage: gawk [POSIX or GNU style options] [--] 'program' file ...
POSIX options:          GNU long options: (standard)
        -f progfile             --file=progfile
        -F fs                   --field-separator=fs
        -v var=val              --assign=var=val
Short options:          GNU long options: (extensions)
        -b                      --characters-as-bytes
        -c                      --traditional
        -C                      --copyright
        -d[file]                --dump-variables[=file]
        -D[file]                --debug[=file]
        -e 'program-text'       --source='program-text'
        -E file                 --exec=file
        -g                      --gen-pot
        -h                      --help
        -i includefile          --include=includefile
        -l library              --load=library
        -L [fatal]              --lint[=fatal]
        -n                      --non-decimal-data
        -M                      --bignum
        -N                      --use-lc-numeric
        -o[file]                --pretty-print[=file]
        -O                      --optimize
        -p[file]                --profile[=file]
        -P                      --posix
        -r                      --re-interval
        -S                      --sandbox
        -t                      --lint-old
        -V                      --version

To report bugs, see node `Bugs' in `gawk.info', which is
section `Reporting Problems and Bugs' in the printed version.
```

```
gawk is a pattern scanning and processing language.
By default it reads standard input and writes standard output.

Examples:
        gawk '{ sum += $1 }; END { print sum }' file
        gawk -F: '{ print $1 }' /etc/passwd
```

awk 内建一些标准变量

变量	说明
FILENAME	文件名
FNR	当前文件的记录数
FS	字段分隔符,默认 " "
NF	当前行的字段数
NR	当前行的编号
OFS	输出的字段分隔符,默认 " "
ORS	输出行的分隔符,默认 "\n"
RS	输入行的分隔符,默认 "\n"

字段

在 awk 的世界里,字段由 $1, $2, $3, ..., $NF 组成。

用 awk 处理文本文件,默认的字段分隔符是空格或者 TAB ,对于 csv 文件,字段一般是用逗号隔开的,因此需要使用参数 -F 来指定分隔符。获取取指定列,比如取第 1 列,第 4 列

```
$ gawk -F ',' '{print $1,$4}' pdr.csv
specid lmjd
104184 55859
102066 55859
105040 55859
104003 55859
107187 55859
113127 55859
106181 55859
110044 55859
102056 55859
```

awk 程序模型

```
pattern { action }
```

pattern 是模式,一般是个判断量或者正则表达式,下面一些例子来说明。

取 lmjd 为 55875 的记录数,选取 specId,lmjd,RA,Dec,class 等列

```
$ gawk -F, '$4==55875 {print $1,$4,$8,$9,$16}' pdr.csv
2415061 55875 31.38040000 59.29930000 STAR
2416046 55875 30.93040000 60.63240000 STAR
```

```
2407073 55875 35.98220000 57.01460000 Unknown
2412135 55875 34.70110000 60.13990000 STAR
2409208 55875 34.51098640 59.03273430 Unknown
2408124 55875 35.30600000 57.58800000 STAR
2414159 55875 30.65200000 59.10650000 STAR
2412220 55875 37.04380000 59.72220000 STAR
2408174 55875 34.25460000 57.80990000 STAR
2407135 55875 36.75950000 57.20690000 STAR
```

模式可以是一个判断量,也可以是多个,多个之间可以使用 `&& ||` 等逻辑运算符,比如:

取 lmjd 为 55875、且 class 为 STAR 的记录数,选取 specId,lmjd,RA,Dec,class 等列

```
$ gawk -F, '($4==55875) && ($16=="STAR") {print $1,$4,$8,$9,$16}' pdr.csv
2415061 55875 31.38040000 59.29930000 STAR
2416046 55875 30.93040000 60.63240000 STAR
2412135 55875 34.70110000 60.13990000 STAR
2408124 55875 35.30600000 57.58800000 STAR
2414159 55875 30.65200000 59.10650000 STAR
2412220 55875 37.04380000 59.72220000 STAR
2408174 55875 34.25460000 57.80990000 STAR
2407135 55875 36.75950000 57.20690000 STAR
2408190 55875 34.30710000 57.33340000 STAR
2406243 55875 35.80650000 57.75390000 STAR
```

还有一个命令 `cut` 也可以获取选定的字段列,但功能较简单。

```
cut -c list [file ...]
cut -f list [-s] [-d delim] [file ...]
```

取第 1、2、3、8、9 列

```
$ cut -d, -f 1-3,8,9 pdr.csv
specid,designation,obsdate,ra,dec
104184,J220716.38+000441.5,2011-10-24,331.81827000,0.07820000
102066,J220545.34-012035.0,2011-10-24,331.43893000,-1.34306000
105040,J220800.83-003953.3,2011-10-24,332.00348000,-0.66481000
104003,J220832.29+002951.0,2011-10-24,332.13455000,0.49751000
107187,J221403.28-005910.0,2011-10-24,333.51370000,-0.98613000
113127,J221652.14+005312.8,2011-10-24,334.21729000,0.88689000
106181,J221552.14-002117.4,2011-10-24,333.96725000,-0.35486000
110044,J220142.30-002458.6,2011-10-24,330.42627000,-0.41628000
102056,J220652.19-010929.6,2011-10-24,331.71747000,-1.15823000
```

`awk` 程序可以做很多很有意思的事情, 比如 `LAMOST` 光谱的文件名定义是: `spec-lmjd-planId_spId_fiberId.fits` ,例如 `spec-55859-F5902_sp11-147.fits` 。我们现在从 pdr.csv 里面的几个字段来拼接这个文件名。

```
$ head pdr.csv | gawk -F, 'NR>1 {printf("spec-%d-%s_sp%02d-%03d.fits\n", $4,$5,$6,$7)
spec-55859-F5902_sp04-184.fits
spec-55859-F5902_sp02-066.fits
spec-55859-F5902_sp05-040.fits
spec-55859-F5902_sp04-003.fits
spec-55859-F5902_sp07-187.fits
spec-55859-F5902_sp13-127.fits
spec-55859-F5902_sp06-181.fits
spec-55859-F5902_sp10-044.fits
spec-55859-F5902_sp02-056.fits
```

检查字段数有时候文件里有些字段是缺失的,那么在一些程序读入文件时会报错,可以使用 awk 程序来检查文件的字段数。比如对于 pdr.csv 。

查看字段数

```
$ head -n 2 pdr.csv | gawk -F, '{print NF}'
34
34
```

pdr.csv 文件的字段数是 34,检查所有的行,如果行数不等于 34,输出行数

```
$ gawk -F, '(NF != 34) {print NR, $0}' pdr.csv
```

无输出,则表明所有行的字段数都是 34;如果有输出的话,会输出行号,和行的全部内容,$0 标识全行。

9.4 文本替换

文本替换也一些数据的标准化工作中也有用,比如在文本中出现一些特殊字符,需要将其转换为数字,就可以使用 sed 进行文本的替换。

```
sed script [-n] script [file ...]
sed [-n] [-e script] [file ...]

sed 's/origin/dest/g'
```

将文本中的 N 替换为 9999.99

```
$ sed 's/\N/9999.99/g'
```

如果只替换第一个 \N ,则不需要参数 g 。

9.5 文件排序

日常环境下,还需要对文件进行一些排序,我们这里将使用 sort 命令来完成这个操作。

```
Usage: sort [OPTION]... [FILE]...
Write sorted concatenation of all FILE(s) to standard output.

Mandatory arguments to long options are mandatory for short options too.
Ordering options:

  -b, --ignore-leading-blanks  ignore leading blanks
  -d, --dictionary-order      consider only blanks and alphanumeric characters
  -f, --ignore-case           fold lower case to upper case characters
  -g, --general-numeric-sort  compare according to general numerical value
  -i, --ignore-nonprinting    consider only printable characters
  -M, --month-sort            compare (unknown) < `JAN' < ... < `DEC'
  -n, --numeric-sort          compare according to string numerical value
  -r, --reverse               reverse the result of comparisons

Other options:

  -c, --check               check whether input is sorted; do not sort
  -k, --key=POS1[,POS2]     start a key at POS1, end it at POS2 (origin 1)
  -m, --merge               merge already sorted files; do not sort
  -o, --output=FILE         write result to FILE instead of standard output
  -s, --stable              stabilize sort by disabling last-resort comparison
  -S, --buffer-size=SIZE    use SIZE for main memory buffer
  -t, --field-separator=SEP  use SEP instead of non-blank to blank transition
  -T, --temporary-directory=DIR  use DIR for temporaries, not $TMPDIR or /tmp;
                              multiple options specify multiple directories
  -u, --unique              with -c, check for strict ordering;
                              without -c, output only the first of an equal run
  -z, --zero-terminated     end lines with 0 byte, not newline
      --help     display this help and exit
      --version  output version information and exit

POS is F[.C][OPTS], where F is the field number and C the character position
in the field.  OPTS is one or more single-letter ordering options, which
override global ordering options for that key.  If no key is given, use the
entire line as the key.

SIZE may be followed by the following multiplicative suffixes:
% 1% of memory, b 1, K 1024 (default), and so on for M, G, T, P, E, Z, Y.

With no FILE, or when FILE is -, read standard input.
```

默认情况下,sort 会对文件按第一列进行排序,与 awk 类似,默认文件字段的分隔符是空格,csv 文件需要指定分隔符。我们生成了一个测试文件:demo.csv。

默认排序默认排序使用的是字母顺便排序,非数字排序。

```
$ sort -t, demo.csv
10914144,55899,GAC_045N28_B1,43.91125400,29.17345800,STAR
1110061,55862,M6203,43.76999500,27.94043700,STAR
11402191,55903,B90306,151.42129100,25.98998100,STAR
1315248,55863,B6302,339.62455500,31.70643000,STAR
13904007,55907,B90703,62.49372000,45.60019500,STAR
1510146,55863,GAC_105N29_B1,102.37585000,29.52028300,STAR
15409061,55910,B91005,166.20455300,28.16800900,STAR
15812186,55910,M31_007N34_B2,14.48237300,36.65349100,STAR
16211165,55911,GAC_082N27_B1,86.99437900,29.84934000,STAR
17516054,55914,GAC_04h29_B1,60.22097000,30.74489900,STAR
18213065,55915,F5591504,123.66046700,15.27538200,Unknown
```

按数字对第 1 列排序参数 -n 表示使用数字进行排序。

```
$ sort -t, -n demo.csv
212082,55859,F5907,46.14507700,0.89738800,STAR
1110061,55862,M6203,43.76999500,27.94043700,STAR
1315248,55863,B6302,339.62455500,31.70643000,STAR
1510146,55863,GAC_105N29_B1,102.37585000,29.52028300,STAR
1907236,55874,GAC_097N28_B1,98.61692100,26.16636000,STAR
3213162,55876,GAC_089N28_B2,90.55748000,29.84290700,STAR
4609224,55879,B7905_2,34.84710000,59.13210000,Unknown
4711016,55880,B8001_1,345.50746030,30.07419800,STAR
4910057,55880,B8004_1,38.53449400,53.57540700,STAR
7107093,55886,M31_004437N404045_M2,13.22132000,39.05297300,STAR
7415091,55889,F8906,54.52611700,8.55509900,STAR
7705015,55890,B9002,15.89809400,28.69114100,Unknown
7708134,55890,B9002,16.76051200,29.16503500,STAR
8305131,55892,F9204,51.18479000,3.83121000,Unknown
8408109,55892,F9205,123.72334000,5.19157400,STAR
8506162,55892,GAC_082N27_M1,90.15996400,26.77842600,STAR
```

按第二列进行排序按列排序,使用参数 -k n

```
$ sort -t, -n -k 2 demo.csv
212082,55859,F5907,46.14507700,0.89738800,STAR
1110061,55862,M6203,43.76999500,27.94043700,STAR
1315248,55863,B6302,339.62455500,31.70643000,STAR
1510146,55863,GAC_105N29_B1,102.37585000,29.52028300,STAR
1907236,55874,GAC_097N28_B1,98.61692100,26.16636000,STAR
```

```
3213162,55876,GAC_089N28_B2,90.55748000,29.84290700,STAR
4609224,55879,B7905_2,34.84710000,59.13210000,Unknown
4711016,55880,B8001_1,345.50746030,30.07419800,STAR
4910057,55880,B8004_1,38.53449400,53.57540700,STAR
7107093,55886,M31_004437N404045_M2,13.22132000,39.05297300,STAR
7415091,55889,F8906,54.52611700,8.55509900,STAR
7705015,55890,B9002,15.89809400,28.69114100,Unknown
7708134,55890,B9002,16.76051200,29.16503500,STAR
8305131,55892,F9204,51.18479000,3.83121000,Unknown
8408109,55892,F9205,123.72334000,5.19157400,STAR
```

按多列排序,按第 3,4 列排序

```
$sort -t, -n -k 3,4 demo.csv
19906159,55919,B5591903,45.55115200,53.20690200,STAR
20514192,55920,B5592004,171.42438900,30.81533100,STAR
20505019,55920,B5592004,173.76449000,29.14064000,STAR
21816123,55922,B5592204,143.58729000,32.87690000,STAR
22003066,55922,B5592206,175.75328100,28.57599400,STAR
22001057,55922,B5592206,176.47797270,26.53900900,STAR
25403087,55930,B5593002,39.64690550,54.19311280,STAR
25508027,55930,B5593003,63.93560400,45.33885900,STAR
28509207,55939,B5593906,183.03336800,31.49157000,STAR
29208170,55940,B5594004,139.85126200,30.79254500,STAR
29314229,55940,B5594006,165.46099660,27.05567000,STAR
29403213,55940,B5594007,183.56564600,30.64137500,STAR
34008035,55954,B5595403,138.08281000,29.70588100,STAR
```

其他的开关指令:

- `-r` 逆序
- `-u` 有重复行的只取首行

uniq 是一个去重的程序,如果在文本文件中,有些行是重复的,可以使用这个程序进行去重,去重前必须先进行排序,也就是先使用 `sort` 处理完文本,然后使用 `uniq` 进行去重,中间可以通过管道连接。

```
sort file | uniq > outfile
```

9.6 文件统计

`wc` 可以用来统计一个文本文件的行数、字数以及字符数。

统计行数

```
$ wc -l pdr.csv
717497 pdr.csv
```

统计字数

```
$ wc -w pdr.csv
801716 pdr.csv
```

统计字节数

```
$ wc -c pdr.csv
156645054 pdr.csv
```

```
$ wc pdr.csv
717497  801716 156645054 pdr.csv
```

9.7 pandas

pandas 在数据的操作方面非常强大,可以参考其文档获得帮助。

http://pandas.pydata.org/pandas-docs/stable/index.html

CHAPTER 10

FITS 文件

10.1 FITS 文件校验

在这里使用安装 astropy 后新增的 2 个 fits 相关软件

10.1.1 fitscheck

FITS 检查工具:

```
Usage: fitscheck [options] <.fits files...>

        .e.g. fitscheck example.fits

        Verifies and optionally re-writes the CHECKSUM and DATASUM keywords
        for a .fits file.
        Optionally detects and fixes FITS standard compliance problems.

Options:
  -h, --help            show this help message and exit
  -k [standard | nonstandard | either | none], --checksum=[standard | nonstandard | e
                        Choose FITS checksum mode or none.  Defaults standard.
  -w, --write           Write out file checksums and/or FITS compliance fixes.
  -f, --force           Do file update even if original checksum was bad.
  -c, --compliance      Do FITS compliance checking; fix if possible.
  -i, --ignore-missing  Ignore missing checksums.
  -v, --verbose         Generate extra output.
```

这也是一个验证 FITS 文件是否符合标准的工具,由于版本的演进和历史的原因,有些历史的 FITS 文件并不符合现代的最新标准。

```
$ fitscheck pdr.fits
MISSING 'pdr.fits' .. Checksum not found in HDU #0
1 errors
```

10.1.2 fitsdiff

FITS 图像比较工具:

```
Usage
=====
  Compare two FITS image files and report the differences in header keywords and
data.

    fitsdiff [options] filename1 filename2

where filename1 filename2 are the two files to be compared.  They may also be
wild cards, in such cases, they must be enclosed by double or single quotes, or
they may be directory names.  If both are directory names, all files in each of
the directories will be included; if only one is directory name, then the
directory name will be prefixed to the file name(s) specified by the other
argument.  for example::

    fitsdiff "*.fits" "/machine/data1"

will compare all FITS files in the current directory to the corresponding files
in the directory /machine/data1.
```

10.2 Funtools

FITS 文件的基本操作,我们也可以通过一些基本的命令行来完成,这里使用的是 Funtools, Funtools 包含了多个简单工具可以帮我们来实现对 FITS 文件的操作。

- **funcalc**: Funtools calculator (for binary tables)
- **funcen**: find centroid (for binary tables)
- **funcnts**: count photons in specified regions
- **funcone**: cone search on RA, Dec columns
- **fundisp**: display data in a Funtools data file
- **funhead**: display a header in a Funtools file
- **funhist**: create a 1D histogram of a column
- **funimage**: create a FITS image from a Funtools data file
- **funindex**: create an index on a column in a binary table

- **funjoin**: join two or more FITS binary tables on specified columns
- **funmerge**: merge one or more Funtools table files
- **funsky**: convert between image and sky coordinates, using WCS info from a FITS header
- **funtable**: copy selected rows from a Funtools file to a FITS binary table
- **funtbl**: extract a table from Funtools ASCII output

10.2.1 funhead

`funhead` 用来显示 FITS 文件的头信息,默认显示 `HDU` ,可以加上 `extname` 或者编号显示不同的部分。

```
usage: funhead [-a] [-s] [-t] [-L] iname[ext] [oname ename]
optional switches:
  -a  # display all extension headers
  -s  # display 79 chars instead of 80 before the new-line
  -t  # prepend data type char to each line of output
  -L  # output in rdb/starbase list format
```

显示 LAMOST 一条光谱的头信息

```
$ funhead spec-55859-F5902_sp11-147.fits
SIMPLE  =                    T /Primary Header created by MWRFITS v1.8
BITPIX  =                  -32 /
NAXIS   =                    2 /
NAXIS1  =                 3908 /
NAXIS2  =                    5 /
EXTEND  =                    T /Extensions may be present
COMMENT --------FILE INFORMATION
FILENAME= 'spec-55859-F5902_sp11-147.fits' /
AUTHOR  = 'LAMOST Pipeline'    / Who compiled the information
N_EXTEN =                    1 / The extension number
EXTEN0  = 'Flux Inverse Subcontinuum Andmask Ormask' /
ORIGIN  = 'NAOC-LAMOST'        / Organization responsible for creating this file
DATE    = '2013-11-13T01:37:57' / Time when this HDU is created (UTC)
COMMENT --------TELESCOPE PARAMETERS
TELESCOP= 'LAMOST  '           / GuoShouJing Telescope
LONGITUD=              117.580 / [deg] Longitude of site
LATITUDE=              40.3900 / [deg] Latitude of site
FOCUS   =                19964 / [mm] Telescope focus
CAMPRO  = 'NEWCAM  '           / Camera program name
CAMVER  = 'v2.0    '           / Camera program version
COMMENT --------OBSERVATION PARAMETERS
DATE-OBS= '2011-10-24T12:11:56.13' / The observation median UTC
```

```
DATE-BEG= '2011-10-24T19:34:44.0' / The observation start local time
DATE-END= '2011-10-24T21:32:02.0' / The observation end local time
LMJD    =                55859 / Local Modified Julian Day
MJD     =                55860 / Modified Julian Day
LMJMLIST= '80436692-80436737-80436775' / Local Modified Julian Minute list
PLANID  = 'F5902   '           / Plan ID in use
RA      =        331.861420000 / [deg] Right ascension of object
DEC     =        2.53486000000 / [deg] Declination of object
DESIG   = 'LAMOST J220726.74+023205.4' / Designation of LAMOST target
FIBERID =                  147 / Fiber ID of Object
CELL_ID = 'H0613   '           / Fiber Unit ID on the focal plane
X_VALUE =        108.694346250 / [mm] X coordinate of object on the focal plane
Y_VALUE =       -763.107559880 / [mm] Y coordinate of object on the focal plane
OBJNAME = '1237678619553038585' / Name of object
OBJTYPE = 'gal     '           / Object type from input catalog
OBJSOURC= 'SDSS    '           / Name of input catalog
T_INFO  = 'NULL    '           / Target information
T_FROM  = 'SDSS    '           / Target catalog
FIBERTYP= 'Obj     '           / Fiber type of object
MAGTYPE = 'ugriz   '           / Magnitude type of object
MAG1    =                18.81 / [mag] Mag1 of object
MAG2    =                18.51 / [mag] Mag2 of object
MAG3    =                17.94 / [mag] Mag3 of object
MAG4    =                17.51 / [mag] Mag4 of object
MAG5    =                17.33 / [mag] Mag5 of object
MAG6    =                10.00 / [mag] Mag6 of object
MAG7    =                10.00 / [mag] Mag7 of object
OBS_TYPE= 'OBJ     '           / The type of target (OBJ, FLAT, ARC or BIAS)
OBSCOMM = 'Science '           / Science, Test
RADECSYS= 'FK5     '           / Equatorial coordinate system
EQUINOX =              2000.00 / Equinox in years
COMMENT --------SPECTROGRAPH PARAMETERS
SPID    =                   11 / Spectrograph ID
SPRA    =              332.208 / [deg] Average RA of this spectrograph
SPDEC   =              2.37180 / [deg] Average DEC of this spectrograph
SLIT_MOD= 'x2/3    '           / Slit mode, x1 ,x2/3 or x1/2
COMMENT --------WEATHER CONDITION
TEMPCCDB=              -119.10 / [deg] The temperature of blue CCD
TEMPCCDR=              -102.70 / [deg] The temperature of red CCD
SEEING  =                 4.37 / [arcsecond] Seeing during exposure
MOONPHA =                28.30 / [day] Moon phase for a 29.53 days period
TEMP_AIR=                 3.03 / [deg] Temperature outside dome
DEWPOINT=               -12.85 / [deg]
```

```
DUST    = '        '           / Reservation
HUMIDITY=                0.30 /
WINDD   =              126.35 / [deg] Wind direction
WINDS   =                4.45 / [m/s] Wind speed
SKYLEVEL= '        '           / Reservation
COMMENT --------DATA REDUCTION PARAMETERS
LAMPLIST= 'lamphgcdne.dat'     / Arc lamp emission line list
SKYLIST = 'skylines.dat'       / Sky emission line list
EXPID01 = '11b-20111024193838-4-80436692' / ID string for exposure 1
EXPID02 = '11b-20111024202422-5-80436737' / ID string for exposure 2
EXPID03 = '11b-20111024210158-6-80436775' / ID string for exposure 3
EXPID04 = '11r-20111024193439-4-80436692' / ID string for exposure 4
EXPID05 = '11r-20111024202024-5-80436737' / ID string for exposure 5
EXPID06 = '11r-20111024205759-6-80436775' / ID string for exposure 6
NEXP    =                    3 / Number of valid exposures
NEXP_B  =                    3 / Number of valid blue exposures
NEXP_R  =                    3 / Number of valid red exposures
EXPT_B  =              5400.00 / [s] Blue exposure duration time
EXPT_R  =              5400.00 / [s] Red exposure duration time
EXPTIME =              5400.00 / [s] Minimum of exposure time for all cameras
BESTEXP =             80436692 / MJM of the best exposure
SCAMEAN =                 2.97 / [ADU] Mean level of scatter light
EXTRACT = 'aperture'           / Extraction method
SFLATTEN=                    T / Super flat has been applied
PCASKYSB=                    T / PCA sky-subtraction has been applied
NSKIES  =                   22 / Sky fiber number
SKYCHI2 =        2.40777261332 / Mean chi^2 of sky-subtraction
SCHI2MIN=        1.74554401614 / Minimum chi^2 of sky-subtraction
SCHI2MAX=        3.29434358083 / Maximum chi^2 of sky-subtraction
NSTD    =                    5 / Number of (good) standard stars
FSTAR   = '116-182-199-207-248' / Fiber ID of flux standard stars
FCBY    = 'auto    '           / Standard stars origin (auto, manual or catalog)
HELIO   =                    T / Heliocentric correction
HELIO_RV=        24.7575432343 / [km/s] Heliocentric correction
VACUUM  =                    T / Wavelengths are in vacuum
NWORDER =                    2 / Number of linear-log10 coefficients
WFITTYPE= 'LOG-LINEAR'         / Linear-log10 dispersion
COEFF0  =              3.56820 / Central wavelength (log10) of first pixel
COEFF1  =          0.000100000 / Log10 dispersion per pixel
WAT0_001= 'system=linear'      /
WAT1_001= 'wtype=linear label=Wavelength units=Angstroms' /
CRVAL1  =              3.56820 / Central wavelength (log10) of first pixel
CD1_1   =          0.000100000 / Log10 dispersion per pixel
```

```
CRPIX1  =                    1 / Starting pixel (1-indexed)
CTYPE1  = 'LINEAR  '           /
DC-FLAG =                    1 / Log-linear flag
COMMENT --------SPECTRA ANALYSIS RESULTS
VERSPIPE= 'v2.6.4  '           / Version of Pipeline
CLASS   = 'GALAXY  '           / Class of object
SUBCLASS= 'Non     '           / Subclass of object
Z       =       -9999.00000000 / Redshift of object
Z_ERR   =             -9999.00 / Redshift error of object
SN_U    =                 2.30 / SNR of u filter
SN_G    =                 2.72 / SNR of g filter
SN_R    =                 2.97 / SNR of r filter
SN_I    =                 4.92 / SNR of i filter
SN_Z    =                 3.44 / SNR of z filter
END
```

10.2.2 fundisp

`fundisp` 显示 FITS 文件体的内容,可以是图像,也可以是星表。

```
usage: fundisp [-f format] [-l] [-n] [-T] iname [columns|bitpix=n]
optional switches:
  -f 'format' # format definitions for table columns
  -l          # display image as a list containing the columns X, Y, VAL
  -n          # don't output header
  -F [c]      # use specified character as column separator (def: space)
  -T          # output in rdb/starbase format (tab separators)
For tables, columns to display can be specified.
For images, data type (bitpix) of display can be specified.
```

例如

```
$ fundisp pdr.fits[1]
```

10.2.3 funcone

`funcone` 是一个简单的锥形检索工具,针对一个 FITS 星表文件进行操作,输入一个坐标和半径参数,输出是一个新的 FITS 文件,包含了检索结果。

输入坐标和半径可以是 `hms` ,也可以是 `degree` ,也可以是 `rad` ,默认是 `hms` 。也可以是输入一个文件,包含了中心坐标。具体的使用方法见下面的说明。

```
usage: funcone <switches> iname oname ra[hdr] dec[hdr] radius[dr"'] [columns [listcol

  iname:  input FITS file name
```

```
  oname:  output FITS file name
  ra:     RA center position, optional units (def: hours)
  dec:    Dec center position, optional units (def: degrees)
  radius: Search radius, optional units (def: degrees)
  columns: optional columns to output

optional switches:

  -d deccol:[hdr]  # Dec column name, units (def: DEC:d)
  -j               # join columns from list file
  -J               # join columns from list file, output all rows
  -l listfile      # read centers and radii from a list
  -L listfile      # read centers and radii from a list, output list rows
  -n               # don't use cone limits as a filter
  -r  racol:[hdr]  # RA column name, units (def: RA:h)
  -x               # append RA_CEN, DEC_CEN, RAD_CEN, KEY cols
  -X               # append RA_CEN, DEC_CEN, RAD_CEN, KEY cols, output all rows

The default units for input center and radius are:
 ra:hours, dec:degrees, radius: degrees.
Sexagesimal notation is supported.

If a list is specified, then the ra, dec, and radius
can be a list column name or numeric value. Col name
specifiers support the :hdr suffix.

The -n switch can speed up processing of small tables
by skipping the overhead associated with filtering on
the cone limits.

examples:

# defaults: RA col in hours, DEC col in degrees
#           RA pos in hours, DEC pos and radius in degrees
funcone in.fits out.fits 23.45 34.56 0.01
# defaults, but with RA position in degrees:
funcone in.fits out.fits 23.45d 34.56 0.01
# user specified columns in degrees
# RA pos in hms, DEC pos in dms, radius in arcmin:
funcone -r myRa:d -d myDec in.fits out.fits 12:30:15.5 30:12 15'
# take positions from rax and decx columns in a FITS table
# and join columns from list to output
funcone -j -l lst.fits in.fits out.fits 'rax:d' 'decx:d' '0.003:d'
```

例子

检索中心坐标 (330.0,-0.4) ,半径 1° 的区域,输出文件 out1.fits

```
$ funcone -r RA:d -d Dec:d pdr.fits[1] out1.fits 330.0d -0.4d 1.0d
$ fundisp out1.fits[1]
```

10.3 Python FITS 操作

Python 我们使用 astropy.io.fits 来进行读写,下面是几个例子:

```
from astropy.io import fits
```

```
In [4]: header = fits.getheader('pdr.fits')

In [5]: header

Out[5]: SIMPLE  =                    T / conforms to FITS standard
        BITPIX  =                    8 / array data type
        NAXIS   =                    0 / number of array dimensions
        EXTEND  =                    T
        DATE    = '2013-12-08'
        DATAVERS= 'v2.6.4  '
        COMMENT LAMOST Data Center, http://data.lamost.org
        COMMENT EMail: dus@lamost.org
        COMMENT
        COMMENT Chinese Astronomical Data Center(CAsDC), http://casdc.china-vo.org
        COMMENT EMail: casdc@china-vo.org

In [6]: header[1]

Out[6]: 8

In [7]: data = fits.getdata('pdr.fits')

In [8]: data

Out[8]: FITS_rec([ (101001, 'J220848.54-020324.3', '2011-10-24', 55859, 'F5902', 1, 1, 332.20227399999999, -2.0567669999999998, 2.23, 10.69,
        17.99, 23.07, 13.93, 'Star', 'STAR', 'K1', 'ugriz', 18.780001, 17.120001, 16.42, 16.15, 15.97, 99.0, 99.0, 'JF_LEGAS_S', 'Obj', 'SDSS
        _S', 'None', nan, -23.07, 0.0, -23.07, 0.88999999),
               (101002, 'J220953.17-020506.0', '2011-10-24', 55859, 'F5902', 1, 2, 332.47157600000003, -2.0850149999999998, 2.0, 5.52, 14.19,
         20.299999, 14.05, 'Star', 'STAR', 'M0', 'ugriz', 20.91, 18.1, 16.66, 16.049999, 15.67, 99.0, 99.0, 'JF_LEGAS_S', 'Obj', 'SDSS_S', 'N
        one', nan, 27.1, 0.0, 32.049999, 1.35),
               (101008, 'J220928.49-015720.7', '2011-10-24', 55859, 'F5902', 1, 8, 332.36874499999999, -1.9557709999999999, 1.84, 9.9399996,
        25.25, 32.32, 18.290001, 'Star', 'STAR', 'G5', 'ugriz', 18.25, 16.639999, 15.97, 15.77, 15.64, 99.0, 99.0, 'JF_LEGAS_S', 'Obj', 'SDSS
        _S', 'None', nan, 25.040001, 0.0, 25.040001, 0.86000001),
               ...,
               (53216245, 'J153613.70+542303.4', '2012-06-17', 56096, 'VB3_235N52_V2', 16, 245, 234.05710999999999, 54.384295999999999, 1.1,
        3.01, 9.1999998, 11.52, 7.5500002, 'star', 'STAR', 'F5', 'brijhk', 14.54, 12.93, 13.7, 13.15, 12.72, 12.4, 12.37, 'PILOT', 'Obj', 'No
        ne', 'None', nan, 36.150002, 0.0, 36.150002, 2.6900001),
               (53216247, 'J153609.75+542821.1', '2012-06-17', 56096, 'VB3_235N52_V2', 16, 247, 234.04064, 54.472546000000001, 1.3, 4.6300001
        , 11.55, 13.93, 9.8699999, 'star', 'STAR', 'G5', 'brijhk', 15.26, 13.86, 14.27, 13.45, 12.74, 12.29, 12.2, 'PILOT', 'Obj', 'None', 'N
        one', nan, -33.220001, 0.0, -33.220001, 0.72000003),
               (53216250, 'J153624.24+543832.9', '2012-06-17', 56096, 'VB3_235N52_V2', 16, 250, 234.10104000000001, 54.642493999999999, 2.05,
         19.540001, 91.099998, -9999.0, 88.620003, 'star', 'STAR', 'F9', 'brijhk', 11.52, 10.5, 99.0, 10.79, 9.5500002, 9.2299995, 9.1599998,
         'PILOT', 'Obj', 'None', 'None', nan, -45.700001, 0.0, -45.700001, 0.64999998)],
              dtype=[('specId', '>i4'), ('designation', 'S19'), ('obsDate', 'S10'), ('mjd', '>i4'), ('planId', 'S20'), ('spId', 'u1'), ('fibe
        rId', 'u1'), ('RA', '>f8'), ('Dec', '>f8'), ('snru', '>f4'), ('snrg', '>f4'), ('snrr', '>f4'), ('snri', '>f4'), ('snrz', '>f4'), ('ob
        jType', 'S10'), ('class', 'S7'), ('subClass', 'S12'), ('magType', 'S10'), ('mag1', '>f4'), ('mag2', '>f4'), ('mag3', '>f4'), ('mag4',
         '>f4'), ('mag5', '>f4'), ('mag6', '>f4'), ('mag7', '>f4'), ('tsource', 'S10'), ('fiberType', 'S9'), ('tfrom', 'S6'), ('tInfo', 'S20'
        ), ('z', '>f8'), ('RV', '>f4'), ('z_err', '>f8'), ('elodierv', '>f4'), ('elodierv_err', '>f4')])
```

更详细的使用参见:http://docs.astropy.org/en/stable/io/fits/index.html

CHAPTER 11

二进制文件

星表的存储可以是文本格式，比如常用的 CDS 星表格式，还有 FITS 文件存储星表，还有一种星表存储格式是二进制格式存储，典型的是 UCAC 系列星表等。传统的读取二进制星表是使用 C 语言或者 Fortran、甚至 IDL 等。下面介绍的是使用 Python 读取星表，输入文件是 UCAC4 的一个天区文件 z100。

根据星表的说明文件，它的数据结构如下:

```
col byte item   fmt unit       explanation                              notes
------------------------------------------------------------------------------
 1  1- 3 ra     I*4 mas        right ascension at  epoch J2000.0 (ICRS) (1)
 2  5- 8 spd    I*4 mas        south pole distance epoch J2000.0 (ICRS) (1)
 3  9-10 magm   I*2 millimag   UCAC fit model magnitude                  (2)
 4 11-12 maga   I*2 millimag   UCAC aperture  magnitude                  (2)
 5 13    sigmag I*1 1/100 mag  error of UCAC magnitude                   (3)
 6 14    objt   I*1            object type                              (4)
 7 15    cdf    I*1            combined double star flag                (5)
         15 bytes
 8 16    sigra  I*1 mas        s.e. at central epoch in RA (*cos Dec)   (6)
 9 17    sigdc  I*1 mas        s.e. at central epoch in Dec             (6)
10 18    na1    I*1            total # of CCD images of this star
11 19    nu1    I*1            # of CCD images used for this star       (7)
12 20    cu1    I*1            # catalogs (epochs) used for proper motions
          5 bytes
13 21-22 cepra  I*2 0.01 yr    central epoch for mean RA, minus 1900
14 23-24 cepdc  I*2 0.01 yr    central epoch for mean Dec,minus 1900
15 25-26 pmrac  I*2 0.1 mas/yr proper motion in RA*cos(Dec)             (8)
16 27-28 pmdc   I*2 0.1 mas/yr proper motion in Dec
17 29    sigpmr I*1 0.1 mas/yr s.e. of pmRA * cos Dec                   (9)
18 30    sigpmd I*1 0.1 mas/yr s.e. of pmDec                            (9)
```

```
         10 bytes
19 31-34 pts_key I*4           2MASS unique star identifier           (10)
20 35-36 j_m    I*2 millimag   2MASS J  magnitude
21 37-38 h_m    I*2 millimag   2MASS H  magnitude
22 39-40 k_m    I*2 millimag   2MASS K_s magnitude
23 41    icqflg I*1            2MASS cc_flg*10 + ph_qual flag for J    (11)
24 42     (2)   I*1            2MASS cc_flg*10 + ph_qual flag for H    (11)
25 43     (3)   I*1            2MASS cc_flg*10 + ph_qual flag for K_s  (11)
26 44    e2mpho I*1 1/100 mag  error 2MASS J   magnitude               (12)
27 45     (2)   I*1 1/100 mag  error 2MASS H   magnitude               (12)
28 46     (3)   I*1 1/100 mag  error 2MASS K_s magnitude               (12)
         16 bytes
29 47-48 apasm  I*2 millimag   B magnitude from APASS                  (13)
30 49-50  (2)   I*2 millimag   V magnitude from APASS                  (13)
31 51-52  (3)   I*2 millimag   g magnitude from APASS                  (13)
32 53-54  (4)   I*2 millimag   r magnitude from APASS                  (13)
33 55-56  (5)   I*2 millimag   i magnitude from APASS                  (13)
34 57    apase  I*1 1/100 mag  error of B magnitude from APASS         (14)
35 58     (2)   I*1 1/100 mag  error of V magnitude from APASS         (14)
36 59     (3)   I*1 1/100 mag  error of g magnitude from APASS         (14)
37 60     (4)   I*1 1/100 mag  error of r magnitude from APASS         (14)
38 61     (5)   I*1 1/100 mag  error of i magnitude from APASS         (14)
39 62    gcflg  I*1            Yale SPM g-flag*10  c-flag             (15)
         16 bytes
40 63-66 icf(1) I*4            FK6-Hipparcos-Tycho source flag        (16)
41       icf(2) ..             AC2000       catalog match flag        (17)
42       icf(3) ..             AGK2 Bonn    catalog match flag        (17)
43       icf(4) ..             AKG2 Hamburg catalog match flag        (17)
44       icf(5) ..             Zone Astrog. catalog match flag        (17)
45       icf(6) ..             Black Birch  catalog match flag        (17)
46       icf(7) ..             Lick Astrog. catalog match flag        (17)
47       icf(8) ..             NPM  Lick    catalog match flag        (17)
48       icf(9) ..             SPM  YSJ1    catalog match flag        (17)
          4 bytes
49 67    leda   I*1            LEDA galaxy match flag                 (18)
50 68    x2m    I*1            2MASS extend.source flag               (19)
51 69-72 rnm    I*4            unique star identification number      (20)
52 73-74 zn2    I*2            zone number of UCAC2 (0 = no match)    (21)
53 75-78 rn2    I*4            running record number along UCAC2 zone (21)
         12 bytes
--------------------------------------------------------------------------------
```

程序 readU4.py :

```
# -*- coding: utf-8 -*-
import struct

def readU4(file):
    with open(file, 'rb') as f:
        chunk = f.read(78)
        while chunk != "":
            parseRc(chunk)
            chunk = f.read(78)

"""
处理每条记录
rc 为一个 78 字节长的二进制字段
"""
def parseRc(rc):
    rcv = struct.unpack('2I2H1B', rc[0:13])     # 只取部分字节
    print "%d %d %d %d %d" % rcv
```

运行下

```
$ python readU4.py
11083 71317210 15354 16189 60
102609 71354450 15517 15800 6
176594 71319797 16027 16292 5
493677 71852035 15708 15380 9
567777 71517830 15815 16244 99
613374 71549413 16281 16269 99
619425 71986584 15166 15268 99
644773 71584833 16253 16221 27
722583 71966779 16247 16443 99
843015 71982671 14164 14239 9
```

二进制文件读取，关键在 struct 的 unpack 方法上，unpack 按照格式将数据解析出来，`2I2B1B`。

Format	C Type	Python type	Standard size
x	pad byte	no value	
c	char	string of length 1	1
b	signed char	integer	1
B	unsigned char	integer	1
?	_Bool	bool	1
h	short	integer	2
H	unsigned short	integer	2
i	int	integer	4
I	unsigned int	integer	4
l	long	integer	4
L	unsigned long	integer	4
q	long long	integer	8
Q	unsigned long long	integer	8
f	float	float	4
d	double	float	8
s	char[]	string	
p	char[]	string	
P	void *	integer	

Part IV

数据归档

CHAPTER 12

简介

12.1 什么是数据归档?

对于天文数据归档来说,就是以数据为核心,构建一个围绕着数据的全生命周期管理系统,包括有:

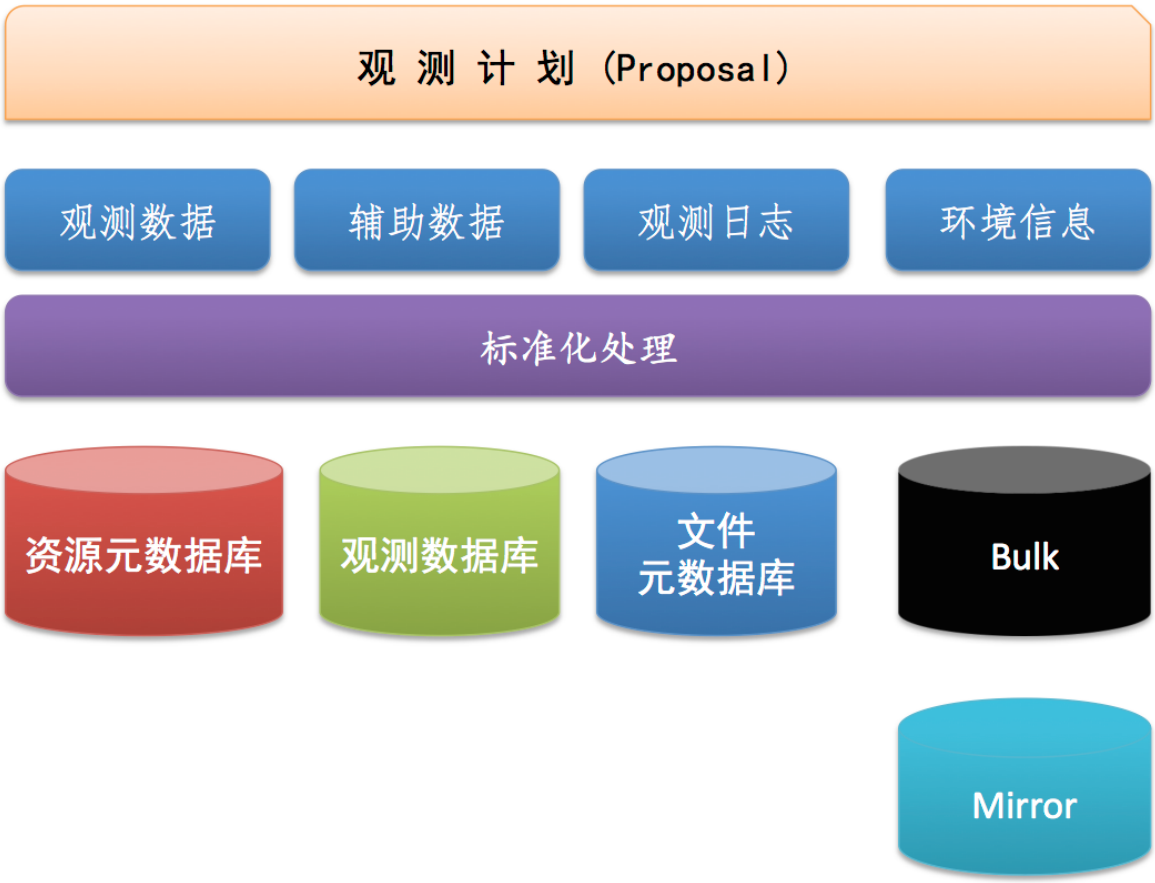
1. 数据生成
2. 数据同化
3. 数据归档
4. 数据处理
5. 数据产品
6. 科研成果

数据归档要服务的对象包括:

1. 观测人员
2. 管理人员
3. 科学用户
4. 公众用户
5. 程序

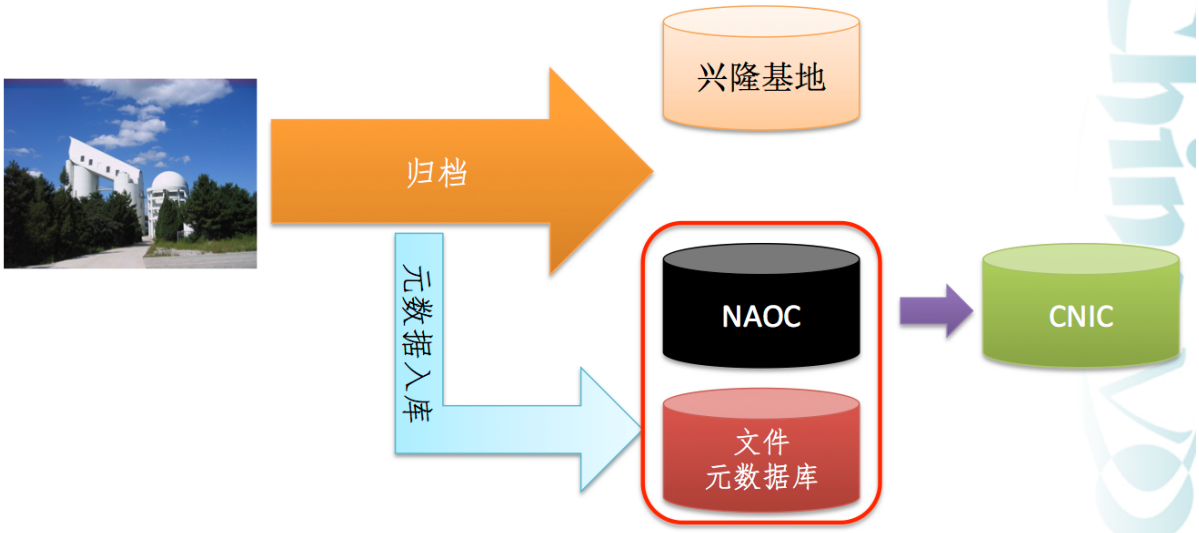
12.2 数据归档

数据归档流程



标准化处理:观测数据、辅助数据、观测日志等

LAMOST 数据传输归档



12.3 元数据

元数据(metadata)是描述数据的数据。比如,描述数据集、描述数据库结构、表结构等。对于 FITS 文件来说,它的头信息可以理解为就是这个 FITS 文件的元数据。数据文件的归档,其重要内容就是用数据库将数据文件管理起来。因此我们就要建设一个数据资源元数据,后面的章节将介绍如何建设数据库? 如何入库? 如何传输? 如何发布?

CHAPTER 13

数据库

本节讲解如何建立一个数据库,并且将 LAMOST PDR 数据导入数据库、建立索引,访问等。

13.1 准备工作

新建数据库用户:

```
$ psql -U postgres template1
psql (9.3.2)
Type "help" for help.

template1=> CREATE ROLE hebl WITH CREATEDB CREATEROLE LOGIN PASSOWRD 'hebl';
```

13.2 建立数据库

建立数据库:

```
$ psql -U hebl template1
template1=> CREATE DATABASE pdr;
template1=> \c pdr
pdr=>
```

导入 pgSphere 库以及用户库

```
$ psql -U postgres pdr < pg_sphere.sql
$ ....
$ psql -U postgres pdr < func.sql
```

func.sql 是一个我们自己定义的一些常用数据库函数:

- `spos(deg, deg)` 返回一个 `spoint` 对象

13.3 建表

建立数据表,我们仍然以 pdr.csv 为例来建立一个数据表。

数据表结构:

```
CREATE TABLE catalogue(
    specId              BIGINT,
    designation         VARCHAR(30),
    obsDate             DATE,
    lmjd                INT4,
    planId              VARCHAR(30),
    spId                SMALLINT,
    fiberId             SMALLINT,
    RA                  NUMERIC(12,8),
    Dec                 NUMERIC(12,8),
    snru                NUMERIC(6,2),
    snrg                NUMERIC(6,2),
    snrr                NUMERIC(6,2),
    snri                NUMERIC(6,2),
    snrz                NUMERIC(6,2),
    objType             VARCHAR(20),
    class               VARCHAR(20),
    subClass            VARCHAR(20),
    magType             VARCHAR(10),
    mag1                NUMERIC(6,2),
    mag2                NUMERIC(6,2),
    mag3                NUMERIC(6,2),
    mag4                NUMERIC(6,2),
    mag5                NUMERIC(6,2),
    mag6                NUMERIC(6,2),
    mag7                NUMERIC(6,2),
    tsource             VARCHAR(16),
    fiberType           VARCHAR(10),
    tfrom               VARCHAR(20),
    tinfo               VARCHAR(64),
    z                   NUMERIC(10,2),
    rv                  NUMERIC(10,2),
    z_err               NUMERIC(8,2),
    elodierv            NUMERIC(10,2),
    elodierv_err        NUMERIC(10,2)
);
```

可以将这个数据表结构存为一个 sql 文件:schema.sql,这样建立一个数据表就可以这样做:

```
$ psql pdr < schema.sql
```

查看建立好的表,以及表结构。

```
pdr=> \d catalogue
             Table "public.catalogue"
    Column    |         Type          | Modifiers
--------------+-----------------------+-----------
 specid       | bigint                | not null
 designation  | character varying(30) |
 obsdate      | date                  |
 lmjd         | integer               |
 planId       | character varying(30) |
 spid         | smallint              |
 fiberid      | smallint              |
 ra           | numeric(12,8)         |
 dec          | numeric(12,8)         |
 snru         | numeric(6,2)          |
 snrg         | numeric(6,2)          |
 snrr         | numeric(6,2)          |
 snri         | numeric(6,2)          |
 snrz         | numeric(6,2)          |
 objtype      | character varying(20) |
 class        | character varying(20) |
 subclass     | character varying(20) |
 magtype      | character varying(10) |
 mag1         | numeric(6,2)          |
 mag2         | numeric(6,2)          |
 mag3         | numeric(6,2)          |
 mag4         | numeric(6,2)          |
 mag5         | numeric(6,2)          |
 mag6         | numeric(6,2)          |
 mag7         | numeric(6,2)          |
 tsource      | character varying(16) |
 fibertype    | character varying(10) |
 tfrom        | character varying(20) |
 tinfo        | character varying(64) |
 z            | numeric(10,2)         |
 rv           | numeric(10,2)         |
 z_err        | numeric(8,2)          |
 elodierv     | numeric(10,2)         |
 elodierv_err | numeric(10,2)         |
 pdr=> \dt
          List of relations
```

```
 Schema |   Name    | Type  | Owner
--------+-----------+-------+-------
 public | catalogue | table | hebl
(1 rows)
```

13.4 数据录入

使用 copy 指令

```
gawk 'NR>1 {print}' pdr.csv |
psql pdr -c "COPY catalogue FROM STDIN USING DELIMITERS ',' NULL ''"

cat pdr.csv | psql pdr -c "COPY catalogue FROM STDIN WITH CSV HEADER"
```

13.5 建立索引

所有字段的索引文件见 index.sql ,可以运行这个 sql 文件:

```
$ psql pdr < index.sql
```

index.sql 文件

```
ALTER TABLE catalogue ADD CONSTRAINT pdr_pk PRIMARY KEY (specId);
CREATE UNIQUE INDEX pdr_idx_spec ON catalogue (planId,lmjd,spId,fiberId);
CREATE INDEX pdr_idx_spos ON catalogue USING GIST(spos(RA,Dec));
CREATE index pdr_idx_designation ON catalogue (designation);
CREATE index pdr_idx_RA ON catalogue (RA);
CREATE index pdr_idx_Dec ON catalogue (Dec);
CREATE index pdr_idx_obsDate ON catalogue (obsDate);
CREATE index pdr_idx_lmjd ON catalogue (lmjd);
CREATE index pdr_idx_planId ON catalogue (planId);
CREATE index pdr_idx_spId ON catalogue (spId);
CREATE index pdr_idx_fiberId ON catalogue (fiberId);
CREATE index pdr_idx_snru ON catalogue (snru);
CREATE index pdr_idx_snrg ON catalogue (snrg);
CREATE index pdr_idx_snrr ON catalogue (snrr);
CREATE index pdr_idx_snri ON catalogue (snri);
CREATE index pdr_idx_snrz ON catalogue (snrz);
CREATE index pdr_idx_objType ON catalogue (objType);
CREATE index pdr_idx_class ON catalogue (class);
CREATE index pdr_idx_subClass ON catalogue (subClass);
CREATE index pdr_idx_magType ON catalogue (magType);
CREATE index pdr_idx_mag1 ON catalogue (mag1);
```

```
CREATE index pdr_idx_mag2 ON catalogue (mag2);
CREATE index pdr_idx_mag3 ON catalogue (mag3);
CREATE index pdr_idx_mag4 ON catalogue (mag4);
CREATE index pdr_idx_mag5 ON catalogue (mag5);
CREATE index pdr_idx_mag6 ON catalogue (mag6);
CREATE index pdr_idx_mag7 ON catalogue (mag7);
CREATE index pdr_idx_tsource ON catalogue (tsource);
CREATE index pdr_idx_tfrom ON catalogue (tfrom);
CREATE index pdr_idx_fiberType ON catalogue (fiberType);
CREATE index pdr_idx_tInfo ON catalogue (tInfo);
CREATE index pdr_idx_z ON catalogue (z);
CREATE index pdr_idx_RV ON catalogue (RV);
CREATE index pdr_idx_elodierv ON catalogue (elodierv);
```

建立好索引后,我们就可以开始使用数据库中的数据。

CHAPTER 14

数据传输

归档完的数据需要从一个地方传输到另一个地方，比如从兴隆观测基地传输到国家天文台数据中心，我们这里依旧使用传统的 Linux 程序来完成这个工作。

14.1 基础

14.1.1 SSH

SSH 是一个远程登录系统，它是 Secure Shell 的缩写，它在我们下面的程序中起一个安全传输的作用。

ssh 一般用法

ssh user@host -p port

默认的 port 是 22，为了安全，建议改成非标准端口，比如 2013。在客户端连接服务端的时候，可以加上一些额外的配置。

.ssh/config

Host astrondata Port 2013

/etc/hosts

192.168.40.28 astrondata

这样的话，访问远程主机的话只需：

```
ssh user@astrondata
```

密钥

ssh 的密钥可以使用我们访问远程主机不需要密码，它的原理是在客户端生成一对密钥，往远程主机的 `.ssh/authorized_keys` 里加入一行客户端的公钥，这样双方就会建立一个自动认证机制。具体做法如何。

生成密钥

```
ssh-keygen -t rsa -C "your_email@example.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/you/.ssh/id_rsa): [Press enter]
Enter passphrase (empty for no passphrase): [Press enter]
Enter same passphrase again:[Press enter]
```

这样会生成一对文件 `id_rsa` 和 `id_rsa.pub` 。前者是私钥,后者是公钥,将后者传输到远程主机上,并加入到 `.ssh/authorized_keys` 文件里

```
cat id_rsa.pub >> .ssh/authorized_keys
```

现在可以尝试下连接,应该可以不需要密码直接连接了。

scp

`scp` 是基于 ssh 一个数据的远程复制程序,类似于 `cp` 进行本地的文件复制操作。

```
usage: scp [-1246BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
           [-l limit] [-o ssh_option] [-P port] [-S program]
           [[user@]host1:]file1 ... [[user@]host2:]file2
```

例:

```
scp -P 2013 file user@astrondata:~/             # 复制文件
scp -P 2013 -r dir user@astrondata:~/   # 复制文件夹
```

如前前面已经配置了 `.ssh/config` 和远程主机的密钥,则上面的可以简化:

```
scp file user@astrondata:~/                     # 复制文件
scp -r dir user@astrondata:~/           # 复制文件夹
```

14.1.2 rsync

`rsync` [1] 是一个开源的数据同步软件,它能同步更新两处计算机的文件与目录,并适当利用差分编码以减少数据传输。rsync 中一项与其他大部分类似程序或协议中所未见的重要特性是镜像对每个目标只需要一次传送。rsync 可拷贝 / 显示目录属性,以及拷贝文件,并可选择性的压缩以及递归拷贝。在常驻模式 (daemon mode) 下,rsync 默认监听 TCP 端口 873。SSH 情况下,rsync 客户端运行程序必须同时在本地和远程机器上安装。我们将使用 SSH 方式进行数据的同步。

rsync 用法

```
rsync is a file transfer program capable of efficient remote update
via a fast differencing algorithm.

Usage: rsync [OPTION]... SRC [SRC]... DEST
  or   rsync [OPTION]... SRC [SRC]... [USER@]HOST:DEST
```

[1] http://rsync.samba.org/

```
  or   rsync [OPTION]... SRC [SRC]... [USER@]HOST::DEST
  or   rsync [OPTION]... SRC [SRC]... rsync://[USER@]HOST[:PORT]/DEST
  or   rsync [OPTION]... [USER@]HOST:SRC [DEST]
  or   rsync [OPTION]... [USER@]HOST::SRC [DEST]
  or   rsync [OPTION]... rsync://[USER@]HOST[:PORT]/SRC [DEST]
The ':' usages connect via remote shell, while '::' & 'rsync://' usages connect
to an rsync daemon, and require SRC or DEST to start with a module name.
```

我们在做传输归档的时候,一般使用的参数是 `-azP` :

选项	说明
`-a`	归档模式,文件或者文件夹都会被传输
`-z`	压缩模式,传输过程中启用压缩
`-P`	输出传输过程的信息

默认情况下,远程的主机端口是 22,如果没有在 `.ssh/config` 中配置端口信息,则可以这么做:

```
rsync -e "ssh -p 2013" -azP src hebl@host:dest
```

将传输过程记录到日志文件

```
rsync -azP src host:dest 2>&1 > tsf_20131212.log
```

14.2 案例

以丽江 2.4 米望远镜观测数据的归档为例,我们建立数据归档与传输:

1. 数据同化: 抽取头信息, 建立元数据表, 除了普通的 FITS 头信息外, 另外加入几个字段
    - `filesize` 文件大小,精确到字节
    - `checksum` 文件校验码,确认文件的完整性
    - `recdate` 入库时间
2. 数据统计:统计此次数据的数目、大小等信息
3. 数据入库
4. 数据传输
5. 发送归档报告

Part V

数据接口

CHAPTER 15

访问数据

本章主要介绍几个访问数据库的技巧和方法。

15.1 psql 访问数据库

我们使用 psql 脚本来访问数据库。

导出一个部分自动到一个 CSV 文件中

```
psql -A -F ',' -t pdr -c "SELECT designation,planId,lmjd,spId,fiberId FROM catalogue"
```

查看下 out.csv

```
$ head out.csv
J031553.29+522639.2,B87806_1,55878,9,13
J032459.64+511215.0,B87806_1,55878,6,120
J031711.05+510142.3,B87806_1,55878,8,108
J030953.95+530028.8,B87806_1,55878,15,197
J031445.85+531551.4,B87806_1,55878,9,121
J030740.45+534510.6,B87806_1,55878,16,122
J030450.51+512832.3,B87806_1,55878,3,245
J031444.68+502315.6,B87806_1,55878,1,103
J032154.18+514025.2,B87806_1,55878,6,82
J031031.82+542218.6,B87806_1,55878,11,144
```

15.2 Python 访问数据库

15.2.1 psycopg2 驱动模式

Python 访问 PostgreSQL 数据库这里使用的驱动时是 psycopg2 [1] , pysycopg2 完全支持 Python DB API 2.0 [2] 。这也是传统的数据库访问模式。

一些例子:

```
>>> import psycopg2
>>> conn = psycopg2.connect("dbname=pdr")
>>> cur = conn.cursor()
>>> q = """SELECT specId,planId,RA,Dec,class FROM catalogue
        WHERE spos(RA,Dec) @ scircle '<(332.00d,-0.664d), 0.2d>'"""
>>> cur.execute(q)
>>> data = cur.fetchall()
>>> print(data)
[(105040L, 'F5902', Decimal('332.00348000'), Decimal('-0.66481000'), 'Unknown'),
 (105137L, 'F5902', Decimal('332.14139000'), Decimal('-0.70628000'), 'Unknown'),
 (105050L, 'F5902', Decimal('331.94492000'), Decimal('-0.80145000'), 'Unknown'),
 (105151L, 'F5902', Decimal('331.95630000'), Decimal('-0.47038000'), 'Unknown'),
 (105028L, 'F5902', Decimal('331.89249000'), Decimal('-0.61879000'), 'GALAXY'),
 (105156L, 'F5902', Decimal('332.14185000'), Decimal('-0.59118000'), 'Unknown'),
 (105041L, 'F5902', Decimal('331.84024000'), Decimal('-0.63403000'), 'Unknown'),
 (105163L, 'F5902', Decimal('332.06512900'), Decimal('-0.50408500'), 'STAR'),
 (105047L, 'F5902', Decimal('331.82643300'), Decimal('-0.75629600'), 'STAR'),
 (105168L, 'F5902', Decimal('332.08339300'), Decimal('-0.51841200'), 'STAR'),
 (105027L, 'F5902', Decimal('331.90488400'), Decimal('-0.70474700'), 'STAR')]
```

15.2.2 pandas

Pandas 有个 SQL 的接口 pandas.io.sql

```
>>> import pandas.io.sql as psql
>>> import psycopg2

>>> conn = psycopg2.connect("dbname=pdr user=hebl")
>>> q = """SELECT specId,planId,RA,Dec,class FROM catalogue
        WHERE spos(RA,Dec) @ scircle '<(332.00d,-0.664d), 0.2d>'"""
>>> data = psql.frame_query(q, conn)
    specid planid          ra       dec    class
0   105040  F5902  332.003480 -0.664810  Unknown
```

[1] http://initd.org/psycopg/
[2] http://www.python.org/dev/peps/pep-0249/

```
1   105137  F5902  332.141390 -0.706280  Unknown
2   105050  F5902  331.944920 -0.801450  Unknown
3   105151  F5902  331.956300 -0.470380  Unknown
4   105028  F5902  331.892490 -0.618790   GALAXY
5   105156  F5902  332.141850 -0.591180  Unknown
6   105041  F5902  331.840240 -0.634030  Unknown
7   105163  F5902  332.065129 -0.504085     STAR
8   105047  F5902  331.826433 -0.756296     STAR
9   105168  F5902  332.083393 -0.518412     STAR
10  105027  F5902  331.904884 -0.704747     STAR
```

15.3 数据下载

有些情况下,我们需要批量下载服务器端的数据,可以使用多种方法进行,这些方法适合不同的场合。

15.3.1 wget

```
Usage: wget [OPTION]... [URL]...
```

`wget` 的参数中

参数	说明
`-c`	断点续传,强烈推荐使用此参数
`-i `` `file`	输入文件,文件的每一行是一个文件的 URL

例如

file

```
http://data.sdss3.org/sas/dr10/apogee/spectro/data/55581/apR-a-00000171.apz
http://data.sdss3.org/sas/dr10/apogee/spectro/data/55581/apR-a-00000172.apz
http://data.sdss3.org/sas/dr10/apogee/spectro/data/55581/apR-a-00000173.apz
http://data.sdss3.org/sas/dr10/apogee/spectro/data/55581/apR-a-00000174.apz
http://data.sdss3.org/sas/dr10/apogee/spectro/data/55581/apR-a-00000175.apz
http://data.sdss3.org/sas/dr10/apogee/spectro/data/55581/apR-a-00000176.apz
http://data.sdss3.org/sas/dr10/apogee/spectro/data/55581/apR-a-00000177.apz
```

下载

```
wget -c -i file
```

15.3.2 curl

```
Usage: curl [options...] <url>
```

curl 与 wget 类似的一个下载工具,但它的功能更为强大,可以发起 POST 。

15.3.3 FTP

FTP 是常用的一个数据下载程序,下面是一个批量下载数据的案例。

FTP 脚本:*get.ftp*

```
open galex.stsci.edu
user anonymous webuser
binary
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-xd-mcat.fits.gz AIS_270_sg28-xd-mcat.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-nd-int.fits.gz AIS_270_sg28-nd-int.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-fd-int.fits.gz AIS_270_sg28-fd-int.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-nd-rrhr.fits.gz AIS_270_sg28-nd-rrhr.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-fd-rrhr.fits.gz AIS_270_sg28-fd-rrhr.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-nd-skybg.fits.gz AIS_270_sg28-nd-skybg.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-fd-skybg.fits.gz AIS_270_sg28-fd-skybg.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-nd-flags.fits.gz AIS_270_sg28-nd-flags.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/AIS_270_sg28-fd-flags.fits.gz AIS_270_sg28-fd-flags.fits.gz
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/qa/manual/AIS_270_sg28-qa.txt AIS_270_sg28-qa.txt
get GR6/pipe/02-vsn/50270-AIS_270/d/01-main/0001-img/07-try/qa/AIS_270_sg28-sumstats.txt AIS_270_sg28-sumstats.txt
bye
```

下载

```
ftp -n -v < get.ftp
```

就可以实现批量下载了。

CHAPTER 16

在线数据访问接口

本节介绍一些有用的小命令行工具来访问网络上的天文数据。

16.1 CDSClient

法国斯特拉斯堡 (CDS) [1] 出品的命令行客户端工具,可以命令行访问其数据和服务。

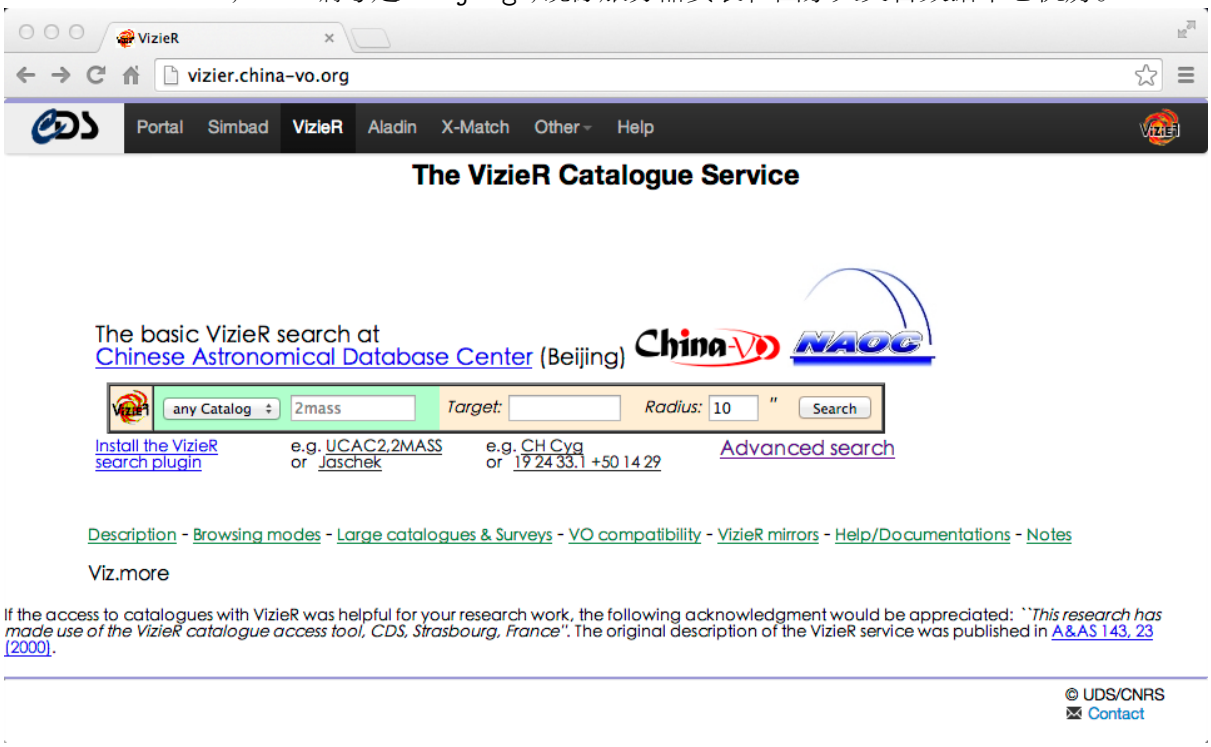
16.1.1 VizieR 检索

vizquery http://vizier.u-strasbg.fr/doc/vizquery.htx

```
Usage: vizquery [-mime={html|asci|vot|vot64|fits|tsv|csv|astrores|xml|acl|text}]
       [-site=site] [{asu_constraints...|input_file_with_contraints}]
  Constraints are given in ASU form (-list can be used for a list of targets)
      vizquery -mime=text -source=I/239/hip_main HIP=1..10
  by default constraints are asked on standard input.
(details at:  http://vizier.u-strasbg.fr/doc/vizquery.htx
   Sites are:
        vizier.u-strasbg.fr          (cds) (fr)
        vizier.cfa.harvard.edu       (cfa) (us)
        vizier.hia.nrc.ca            (cadc) (ca)
        vizier.nao.ac.jp             (adac) (jp)
        vizier.iucaa.ernet.in        (iucaa) (in)
        data.bao.ac.cn               (bejing) (cn)
        vizier.ast.cam.ac.uk         (cambridge) (uk)
        www.ukirt.jach.hawaii.edu    (ukirt) (hawaii)
        vizier.inasan.ru             (moscow) (ru)
```

[1] http://cdsarc.u-strasbg.fr/doc/cdsclient.html

注 国家天文台是其官方的一个镜像点， 域名为: vizier.china-vo.org 或者 data.bao.ac.cn ,site 编号是 beijing ,镜像服务器安装在国家天文台数据中心机房。



-source 是一系列的星表源,这个可以从其网站或者镜像网站上找到。常用的有:

星表	-source
SDSS9	V/139
UCAC4	I/315/out
USNO-B1.0	I/284/out
WISE	II/311/wise
2MASS	II/246/out
Tycho	I/259/tyc2
Hip	I/239/hip_main

constraints 是检索的约束条件,可以参考其网站。

案例

```
$ vizquery -mime=text -source=I/239/hip_main HIP=1..10
#...ASU parameters being sent to vizier
#
#   VizieR Astronomical Server vizier.u-strasbg.fr
#    Date: 2013-12-09T06:21:58 [V1.99+ (14-Oct-2013)]
#   In case of problem, please report to: cds-question@unistra.fr
#
#
#Coosys J2000: eq_FK5 J2000
#INFO votable-version=1.99+ (14-Oct-2013)
#INFO -ref=VOTx25051
#INFO MaxTuples=50000
#INFO queryParameters=3
#-oc.form=D.
#-source=I/239/hip_main
#HIP=1..10
#

#RESOURCE=yCat_1239
#Name: I/239
#Title: The Hipparcos and Tycho Catalogues (ESA 1997)
#Coosys J2000_1991.250: eq_FK5 J2000
#Table I_239_hip_main:
#Name: I/239/hip_main
#Title: The Hipparcos Main Catalogue\vizContent{timeSerie}
#---Details of Columns:
```

```
    HIP             (I6)    Identifier (HIP number) (H1) [ucd=meta.id;meta.main]
    RAhms           (a11)   Right ascension in h m s, ICRS (J1991.25) (H3) [ucd=pos.eq.ra;meta.main]
    DEdms           (a11)   Declination in deg ' ", ICRS (J1991.25) (H4) [ucd=pos.eq.dec;meta.main]
    Vmag (mag)      (F5.2)  ? Magnitude in Johnson V (H5) [ucd=phot.mag;em.opt.V]
    RA(ICRS) (deg) (F12.8) *? alpha, degrees (ICRS, Epoch=J1991.25) (H8) [ucd=pos.eq.ra]
    DE(ICRS) (deg) (F12.8) *? delta, degrees (ICRS, Epoch=J1991.25) (H9) [ucd=pos.eq.dec]
    Plx (mas)       (F7.2)  ? Trigonometric parallax (H11) [ucd=pos.parallax.trig]
    pmRA (mas/yr)   (F8.2)  Proper motion mu_alpha.cos(delta), ICRS(H12) {\em(for J1991.25 epoch)} [ucd=pos.pm;pos.eq.ra]
    pmDE (mas/yr)   (F8.2)  ? Proper motion mu_delta, ICRS (H13) {\em(for J1991.25 epoch)} [ucd=pos.pm;pos.eq.dec]
    e_Plx (mas)     (F6.2)  ? Standard error in Plx (H16) [ucd=stat.error]
    B-V (mag)       (F6.3)  ? Johnson B-V colour (H37) [ucd=phot.color;em.opt.B;em.opt.V]
    Notes           (A1)    *[DGPWXYZ] Existence of notes (H70) [ucd=meta.note]
------ ----------- ----------- ----- ----------- ----------- ------- -------- -------- ------ ------ -
                                                                                                    N
                                                                                                    o
                                                                                                    t
                                      Vmag  RA           DE           Pl      pmRA     pmDE     e_Plx  B-V    e
HIP    RAhms       DEdms          (mag) (ICRS) (deg) (ICRS) (deg) x (mas) (mas/yr) (mas/yr) (mas)  (mag)  s
------ ----------- ----------- ----- ----------- ----------- ------- -------- -------- ------ ------ -
     1 00 00 00.22 +01 05 20.4  9.10   0.00091185   1.08901332    3.54    -5.20    -1.88   1.39  0.482
     2 00 00 00.91 -19 29 55.8  9.27   0.00379737 -19.49883745   21.90   181.21    -0.93   3.10  0.999
     3 00 00 01.20 +38 51 33.4  6.61   0.00500795  38.85928608    2.81     5.24    -2.91   0.63 -0.019
     4 00 00 02.01 -51 53 36.8  8.06   0.00838170 -51.89354612    7.75    62.85     0.16   0.97  0.370
     5 00 00 02.39 -40 35 28.4  8.55   0.00996534 -40.59122440    2.87     2.53     9.07   1.11  0.902
     6 00 00 04.35 +03 56 47.4 12.31   0.01814144   3.94648893   18.80   226.29   -12.84   4.99  1.336
     7 00 00 05.41 +20 02 11.8  9.64   0.02254891  20.03660216   17.74  -208.12  -200.79   1.30  0.740
     8 00 00 06.55 +25 53 11.3  9.05   0.02729160  25.88647445    5.17    19.09    -5.66   1.95  1.102 P
     9 00 00 08.48 +36 35 09.4  8.59   0.03534189  36.58593777    4.81    -6.30     8.42   0.99  1.067
```

```
    10 00 00 08.70 -50 52 01.5  8.59   0.03625309 -50.86707360   10.76    42.23    40.02    1.10   0.489

        #END#  -ref=VOTx25051 ================================================================================
```

16.1.2 find_cats

```
find_cats
```

```
Usage: find_cats file_with_centers [catalog|option]...
   or  find_cats -  [catalog|option]...          (data in stdin)
-----------------------------------------------------------------------
Query one or several catalogues from a list of positions among:
#...cats_server arguments:
2MASS 2MASS6X 2MASSI APM CFHT-LS_D1TO4 CFHT-LS_W1TO4 CMC14 DENIS-P DENIS2
DENIS3 GAIA_GUMS_GAL GAIA_GUMS_STARS GALEX_GR5_AIS GALEX_GR5_MIS GLIMPSE GSC2.2
GSC2.3 GSC_1.1 GSC_1.2 GSC_1.2 GSC_ACT GUMS10_GAL GUMS10_LMC GUMS10_MC
GUMS10_MW GUMS10_MW.V2 GUMS10_SMC IRSF_MCPSC J_AJ_141_189 KIC LMCPS MC2 NOMAD1
OGLE_BULGE PPMX PPMXL SAGE SAGE_ARCH SAGE_ARCH_EP1_EP2 SAGE_CAT_EP1_EP2
SAGE_SMC_ARCH_EP0_EP1_EP2 SAGE_SMC_CAT_EP0_EP1_EP2 SDSS3 SDSS4 SDSS5 SDSS6
SDSS7 SDSS8 SDSS9 SPM4 UCAC1 UCAC2 UCAC3 UCAC4 UKIDSS_DR6_GPS UKIDSS_DR7_LAS
UKIDSS_DR8_DXS UKIDSS_DR8_GCS UKIDSS_DR8_LAS UKIDSS_DR9_DXS UKIDSS_DR9_GCS
UKIDSS_DR9_LAS USNO_A1 USNO_A2 USNO_B1 WISE_ALLSKY WISE_PRELIM
-----------------------------------------------------------------------
#...cats_server arguments: -help
   Besides the file containing positions to query, the arguments include
   catalog(s) name(s) and additional contraints for the positional search.
   The constraints are generic, except when the catalog name is followed
   by a colon (:). The following contraints:
           -r 1 GSC1.2 UCAC3: -r 0.5 2MASS: -lmK 6,13
   ask for a query of catalogues:
   -- GSC1.2 within a target radius of 1arcmin
   -- UCAC3  within a target radius of 0.5arcmin
   -- 2MASS  within a target radius of 1arcmin and K mag in range 6-13.
```

作为一系列特例,`find***` 可以直接访问常用的星表:

```
find2mass    findcat      findgsc1.1   findnomad1   findsdss5    finducac3
find2mass6x  findcmc14    findgsc1.2   findpmm1     findsdss6    finducac4
find2massi   findenis     findgsc1.3   findpmm2     findsdss7    findusnoa1
find_cats    findenis1    findgsc2.2   findppmx     findsdss8    findusnoa2
find_cats.gz findenis2    findgsc2.3   findppmxl    findspm4     findusnob1
find_gen     findenis3    findkic      findsage     findtycab    findwise
find_gen1    findglimpse  findmc2      findsdss3    finducac1    findwisep
findacro     findgsc      findmc_irsf  findsdss4    finducac2    finducac4
```

Usage

```
Usage: findsdss8 center-position [other-options]
   or  findsdss8 Jhhmmss.ss+ddmmss.s    [other-options]
   or  findsdss8 -i SDSS-identification [other-options]
```

```
   or  findsdss8 -f [file_with_centers] [other_options]
   or  findsdss8 - [other_options]      (data in stdin)
-------------------------------------------------------------------
 Several SDSS-identifications may be   Jhhmmss.ss+ddmmss.s
   or   19-digit number           or   run-rerun-camcol-field-obj
   or   plate-mjd-fiber           or   run-camcol-field-obj
-------------------------------------------------------------------
#--- Details on other options:
          [-HELP] [-R root_name] [-r[sd] [min,]radius] [-b[sd] x[,y]]
             [-2] [-z] [-full] [-e edit_opt] [-f input_file] [-m max_records]
             [-c center | -i objID | -run run#] [-l! min,max] [-s !] [-whole]
  -HELP: display column explanations
     -v: verbose option;           -v2 = display the names of the files used
     -b: target box in arcmin ;    -bs = target box in arcsec  -bd in degrees
     -r: target radius in arcmin ; -rs = target radius in arcsec
     -c: target center in decimal or sexagesimal (default in stdin)
     -2: edit also mode=2 sources (mode=1 sources are listed by default)
     -z: restrict to objects with known redshift (in fact with spectroscopy)
     -e: 0=position+mag   a=all    b=basic(default)  i=SDSS-IDs J=Jname+epoch
         p=position_only  m=mas    s=Sexagesimal     x=x,y  ,=CSV   %=parfile
     -f: specifies an input file (default stdin)
    -id: query from SDSS-ID: Jname, 18-digit number or run.camcol.field.obj;
         accept also spectro.ID   : 18-digit number or plate-mjd-fiber
     -m: max number of stars to retrieve
    -l!: Set the limits (range) on one of the parameters (below)
    -s!: Sort the result by the parameter ! (list below)
  -full: shortcut for options -2 (all sources) and -e a (edit all parameters)
 -whole: search on the whole SDSS catalog
     -R: Root (directory) name where the SDSS files are located ($SDSSroot)
====The abbreviations of the parameters (symbolized !) are:
  a=alpha(RA)  c=class(type)  d=delta(Dec)  e=Epoch    e.=errormag
  m.=mag    m.-.=color        q=fieldQual   R=Release  r=distance
  x=projEast   y=projNorth    z=redshift
. represents one of the 5 color bands   u  g  r  i  z
Selection of 'clean photometry' objects: use clean=1 or mode=+
```

检索 SDSS8 星表

```
$ findsdss8 12.23 21.1
#...aclient 130.79.129.161 1660 sdss8 -sr -c 12.23 21.1
#======== SDSS-DR8 server (2012-03, V1.8) ======== CDS, Strasbourg ========
#Center: 12.23 21.1
#+cl SDSS8-Jname          R   RA   (J2000) Dec    e_RA  e_Dec  Obs.Date Q   zsp:e_zsp  u(mag):sigma g(mag):sigma r(mag):sigma i(m
1 3 J004855.27+210553.7  8 012.230307+21.098255 0.224 0.182 2009.0463 3   ---:---   26.124:0.764 24.833:0.798 23.430:0.497 22
1 3 J004855.77+210605.0  8 012.232404+21.101396 0.279 0.240 2009.0463 3   ---:---   22.189:0.377 23.959:0.563 23.174:0.447 22
1 3 J004854.68+210553.3  8 012.227849+21.098159 0.217 0.123 2009.0463 3   ---:---   26.367:0.632 24.293:0.657 22.536:0.244 21
1+3 J004855.02+210615.9  8 012.229268+21.104425 0.066 0.068 2009.0463 3   ---:---   25.564:1.095 22.252:0.116 21.381:0.080 20
1+6 J004854.17+210610.1  8 012.225727+21.102808 0.085 0.075 2009.0463 3   ---:---   24.987:1.438 22.612:0.156 21.683:0.103 21
1+3 J004854.56+210615.2  8 012.227357+21.104238 0.048 0.046 2009.0463 3   ---:---   22.189:0.373 21.478:0.068 20.574:0.045 20
1 3 J004856.13+210540.5  8 012.233914+21.094601 0.182 0.335 2009.0463 3   ---:---   25.643:1.353 24.756:0.974 22.464:0.273 21
1 3 J004856.59+210541.4  8 012.235811+21.094833 0.152 0.113 2009.0463 3   ---:---   26.072:1.050 22.830:0.272 22.246:0.243 20
1 3 J004853.84+210620.6  8 012.224357+21.105725 0.101 0.162 2009.0463 3   ---:---   22.702:0.660 22.655:0.217 21.844:0.158 21
1+6 J004854.17+210628.4  8 012.225734+21.107908 0.109 0.102 2009.0463 3   ---:---   23.113:0.722 21.800:0.078 21.960:0.132 21
1+3 J004852.91+210607.5  8 012.220488+21.102087 0.059 0.053 2009.0463 3   ---:---   22.072:0.530 21.512:0.109 19.961:0.042 19
1 3 J004854.71+210526.7  8 012.227983+21.090749 0.140 0.148 2009.0463 3   ---:---   22.754:0.627 22.642:0.188 22.606:0.269 21
1+3 J004853.45+210623.8  8 012.222746+21.106610 0.176 0.149 2009.0463 3   ---:---   24.693:2.555 24.109:0.920 21.648:0.171 23
```

16.1.3 sesame

名称解析服务 `sesame` [2] ,支持查询 Simbad [3] , NED [4] 和 VizieR [5] 。

用法

```
Usage: /usr/local/bin/sesame [-o{xIfHp}] [-r{SNVA}] [-S{servername}]
         identifier [, identifier...]
The options may be replaced by environment variables:
-o = SESAME_OUTPUT   (default 'x' for XML; )
-r = SESAME_RESOLVER (default SNV for Simbad/Ned/VizieR; A for All)
-S = SESAME_SERVER   (default cds; might be cfa, cadc...)
More details at  http://cds.u-strasbg.fr/doc/sesame.htx
```

例子

```
$ sesame -oI m31
# m31   #Q-00001



#=N=NED:    1
%C G
%J 010.6847929 +41.2690650 = 00:42:44.35 +41:16:08.6
%J.E [80.00 80.00 0] 2010ApJS..189...37E
%V v -300.09229 [ 3.89730] 1991RC3.9.C...0000d
%T SA(s)b         LINER
%MAG  4.36
%I.0 MESSIER 031 =[G]
%I NGC 0224 =[G]
%I Andromeda Galaxy =[G]
%I UGC 00454 =[G]
%I CGCG 535-017 =[G]
%I CGCG 0040.0+4100 =[G]
%I MCG +07-02-016 =[G]
%I GIN 801 =[G]
%I B3 0040+409 =[RadioS]
%I 2MASX J00424433+4116074 =[IrS]
%I IRAS  00400+4059 =[IrS]
%I IRAS F00400+4059 =[IrS]
%I KTG 01C =[G]
%I LDCE 0031 NED007 =[G]
%I HDCE 0029 NED003 =[G]
```

[2] http://cdsarc.u-strasbg.fr/doc/sesame.htx
[3] http://simbad.u-strasbg.fr/simbad
[4] http://ned.ipac.caltech.edu/index.html
[5] http://vizier.u-strasbg.fr/vizier/

```
%I HOLM 017A =[G]
%I PGC 002557 =[G]
%I UZC J004244.3+411608 =[G]
%I 87GB 004002.2+405940 =[RadioS]
%I 87GB[BWE91] 0040+4059 =[RadioS]
%I 6C B004001.6+410004 =[RadioS]
%I MY 0040+409A =[RadioS]
%I CXO J004244.2+411608 =[XrayS]
%I CXO J004244.3+411608 =[XrayS]
%I CXOM31 J004244.3+411608 =[XrayS]
%I RX J0042.6+4115 =[XrayS]
%I 1RXS J004241.8+411535 =[XrayS]
%I MAXI J0043+410 =[XrayS]
%I 2PBC J0042.6+4111 =[XrayS]
%I XMMLPt 1010 =[XrayS]
%I XMMM31 J004244.1+411607 =[XrayS]
%I 2XMM J004244.5+411611 =[XrayS]
%I 2XMMp J004244.4+411612 =[XrayS]
%I EXSS 0039.9+4059 =[XrayS]
%I 1H 0039+408 =[XrayS]
%I 1ES 0039+409 =[XrayS]
%I XSS J00425+4102 =[XrayS]
%I 2FGL J0042.5+4114 =[GammaS]
%I [PFJ93] 44 =[XrayS]
%I [SPB93] 010 =[G]
%I LGG 011:[G93] 001 =[G]
%I [MHH96] J004241+411531 =[XrayS]
%I [VCV2001] J004244.3+411610 =[G]
%I MESSIER 031:[KGP2002] r1-010 =[V*]
%I MESSIER 031:[PFH2005] 321 =[XrayS]
%I MESSIER 031:[VG2007] 001 =[XrayS]
%I 0039+408 =[Other]
%I 0040+4059 =[Other]
%I LEDA 002557 =[G]
#B 3561


#====Done (2013-Dec-13,07:39:46z)====
```

16.2 astroquery

`astroquery` [6] 是 `astropy` 家族的一个程序包,它包含了一组访问在线天文数据的工具。

16.2.1 工具集

- **Simbad**: Basic data, cross-identifications, bibliography and measurements for astronomical objects outside the solar system.
- **Vizier**: Set of 11,000+ published, multiwavelength catalogues hosted by the CDS.
- **IRSA** dust: Galactic dust reddening and extinction maps from IRAS 100 um data.
- **NED**: NASA/IPAC Extragalactic Database. Multiwavelength data from both surveys and publications.
- **IRSA**: NASA/IPAC Infrared Science Archive. Science products for all of NASA's infrared and sub-mm missions.
- **UKIDSS**: UKIRT Infrared Deep Sky Survey. JHK images of 7500 sq deg. in the northern sky.
- **MAGPIS**: Multi-Array Galactic Plane Imaging Survey. 6 and 20-cm radio images of the Galactic plane from the VLA.
- **NRAO**: Science data archive of the National Radio Astronomy Observatory. VLA, JVLA, VLBA and GBT data products.
- **Besancon**: Model of stellar population synthesis in the Galaxy.
- **NIST**: National Institute of Standards and Technology (NIST) atomic lines database.
- **Fermi**: Fermi gamma-ray telescope archive.
- **SDSS**: Sloan Digital Sky Survey data, including optical images, spectra, and spectral templates.
- **Alfalfa**: Arecibo Legacy Fast ALFA survey; extragalactic HI radio data.
- **SHA**: Spitzer Heritage Archive; infrared data products from the Spitzer Space Telescope
- **Lamda**: Leiden Atomic and Molecular Database; energy levels, radiative transitions, and collisional rates for astrophysically relevant atoms and molecules.
- **Ogle**: Optical Gravitational Lensing Experiment III; information on interstellar extinction towards the Galactic bulge.
- **Splatalogue**: National Radio Astronomy Observatory (NRAO)-maintained (mostly) molecular radio and millimeter line list service.

[6] http://www.astropy.org/astroquery/

16.2.2 案例

```
>>> from astroquery.simbad import Simbad
>>> result_table = Simbad.query_object("m31")
>>> result_table.pprint(show_unit=True)
MAIN_ID      RA           DEC      RA_PREC DEC_PREC COO_ERR_MAJA COO_ERR_MINA COO_ERR_
          "h:m:s"      "d:m:s"                          mas           mas           deg
------- ------------ ------------ ------- -------- ------------ ------------ --------
  M  31 00 42 44.330 +41 16 07.50       7        7          nan          nan
```

更多案例:http://astroquery.readthedocs.org/en/latest/

16.3 SDSS DR10

SDSS DR10 [7] 今年发布的网站中也提供了一种数据访问的接口 [8] :比如图像访问、光谱访问和红外光谱访问。

16.3.1 图像访问

根据 RUN ,CAMCOL 和 FIELD 三个选项来确定一幅图像。

API

```
http://data.sdss3.org/fields/runCamcolField?run=RUN&camcol=CAMCOL&field=FIELD
```

比如:

```
http://data.sdss3.org/fields/runCamcolField?field=187&camcol=3&run=3712
```

16.3.2 光谱访问

API

```
http://data.sdss3.org/spectrumDetail?plateid=PLATEID&mjd=MJD&fiber=FIBERID
```

16.3.3 红外光谱访问

API

```
http://data.sdss3.org/irSpectrumDetail?plateid=PLATEID&mjd=MJD&fiber=FIBERID
```

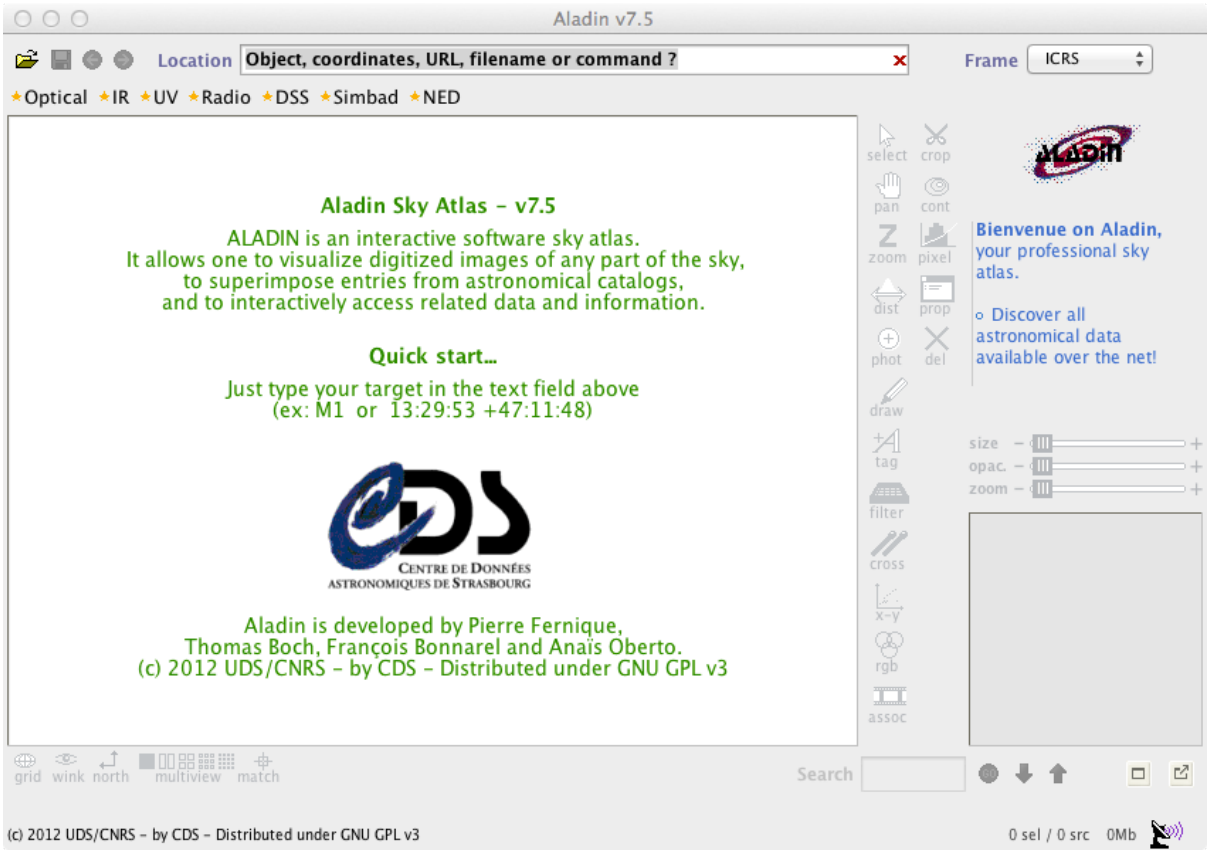
[7] http://dr10.sdss3.org/
[8] http://dr10.sdss3.org/documentation

Part VI

VO 程序

CHAPTER 17

Aladin



Aladin 是法国斯特拉斯堡天文数据中心 (CDS) [1] 开发的一款 VO 软件，目前最新版是 `v7.5` ，它可以展示星图、可以可视化星表，最重要的是它也可以被当成一个数据终端，通过 `Aladin` 可以获取数十个数据中心的数据。完全可以将其作为一个星表、星图数据的检索下载终端。

完整的 Aladin 用户手册请访问：http://aladin.u-strasbg.fr/java/AladinManual6.pdf

在线演示：http://aladin.u-strasbg.fr/java/DemoAladinV7.htm

[1] Centre de Données astronomiques de Strasbourg(CDS): http://cds.u-strasbg.fr/

17.1 使用案例

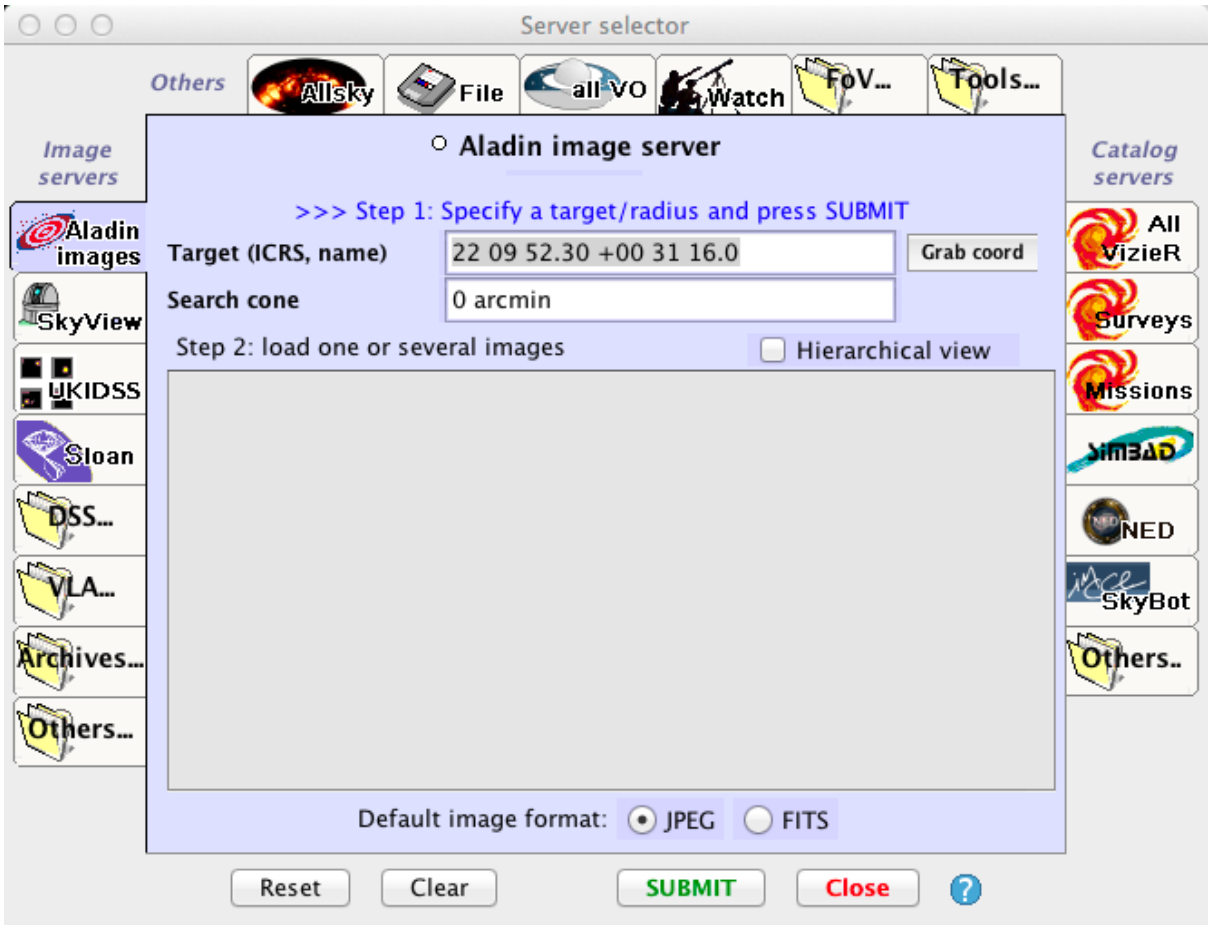


Figure 17.1: 打开文件,可以输入坐标参数,可以选择从不同的源检索数据

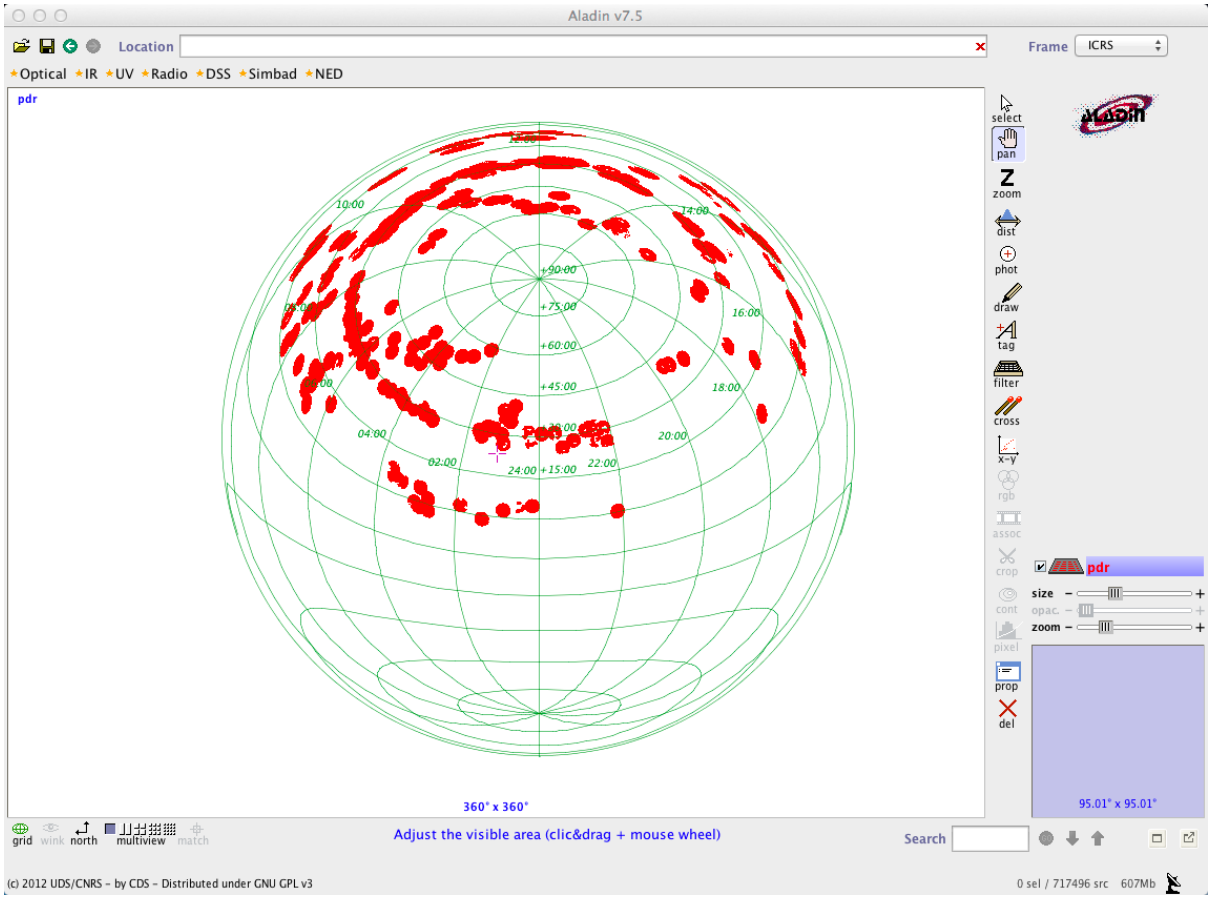
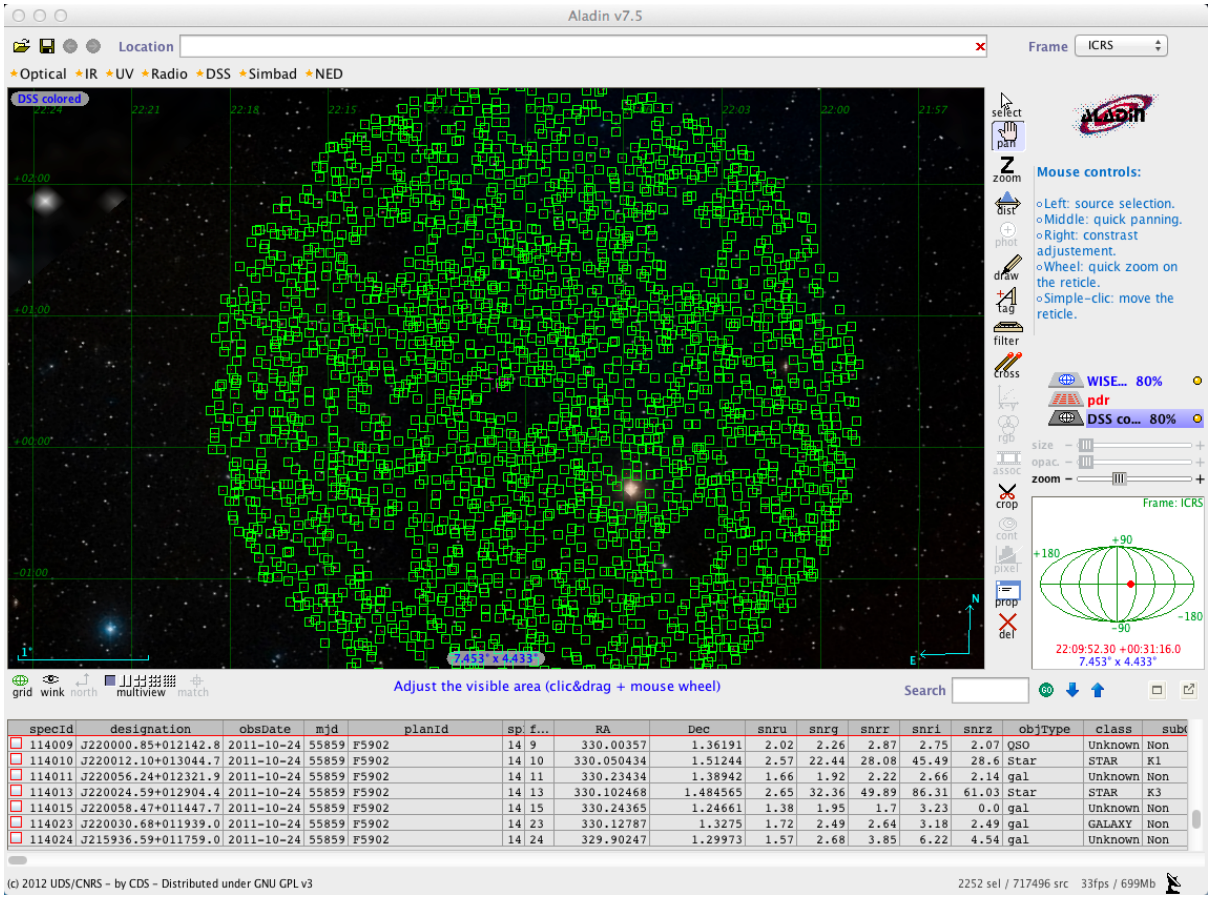
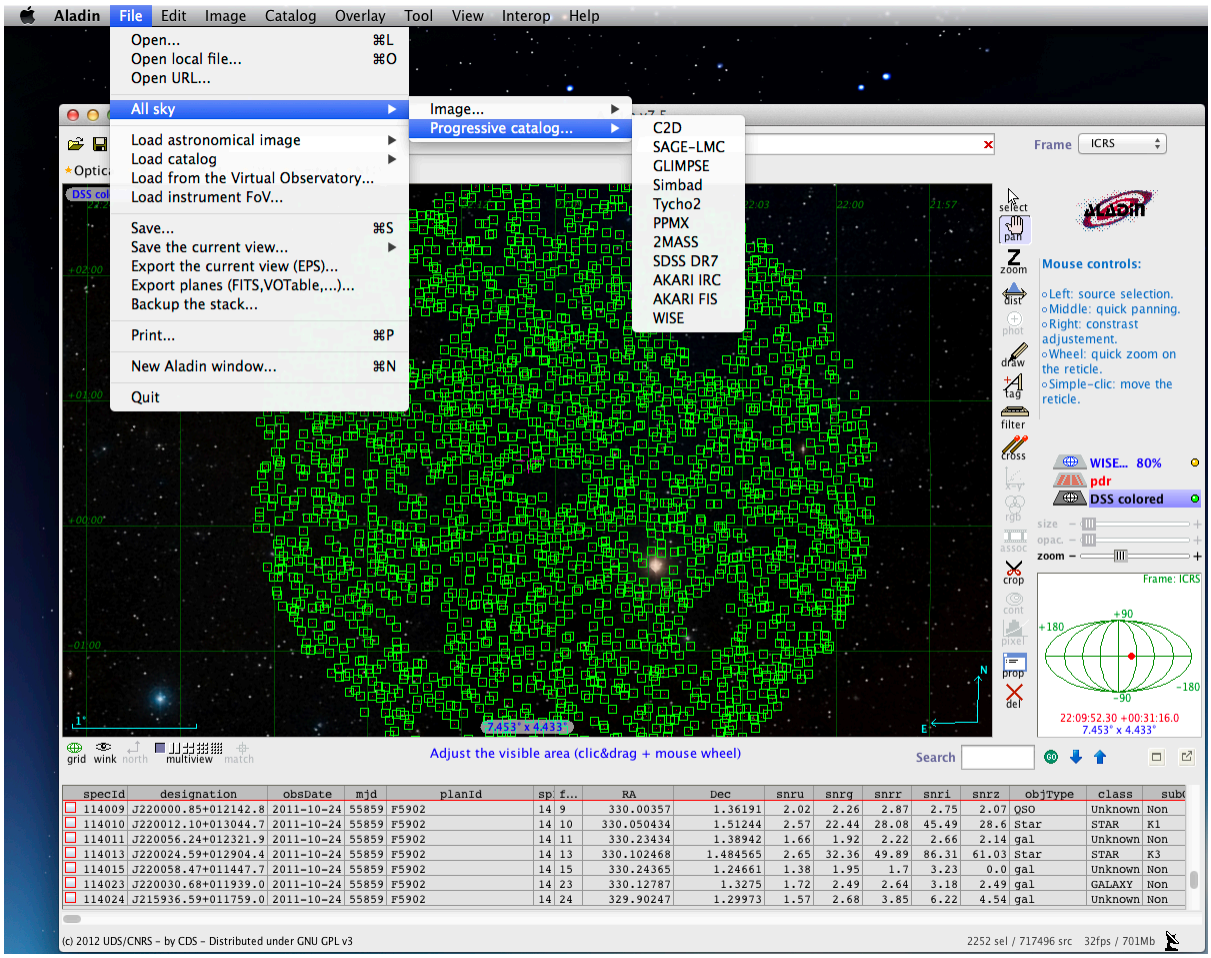


Figure 17.2: 打开本地的 pdr.fits 文件



specId	designation	obsDate	mjd	planId	sp	f...	RA	Dec	snru	snrg	snrr	snri	snrz	objType	class	sub
114009	J220000.85+012142.8	2011-10-24	55859	F5902	14	9	330.00357	1.36191	2.02	2.26	2.87	2.75	2.07	QSO	Unknown	Non
114010	J220012.10+013044.7	2011-10-24	55859	F5902	14	10	330.050434	1.51244	2.57	22.44	28.08	45.49	28.6	Star	STAR	K1
114011	J220056.24+012321.9	2011-10-24	55859	F5902	14	11	330.23434	1.38942	1.66	1.92	2.22	2.66	2.14	gal	Unknown	Non
114013	J220024.59+012904.4	2011-10-24	55859	F5902	14	13	330.102468	1.484565	2.65	32.36	49.89	86.31	61.03	Star	STAR	K3
114015	J220058.47+011447.7	2011-10-24	55859	F5902	14	15	330.24365	1.24661	1.38	1.95	1.7	3.23	0.0	gal	Unknown	Non
114023	J220030.68+011939.0	2011-10-24	55859	F5902	14	23	330.12787	1.3275	1.72	2.49	2.64	3.18	2.49	gal	GALAXY	Non
114024	J215936.59+011759.0	2011-10-24	55859	F5902	14	24	329.90247	1.29973	1.57	2.68	3.85	6.22	4.54	gal	Unknown	Non

Figure 17.3: 可以载人 DSS 星图、WISE 星表,并且叠加



specId	designation	obsDate	mjd	planId	sp	f...	RA	Dec	snru	snrg	snrr	snri	snrz	objType	class	sub
114009	J220000.85+012142.8	2011-10-24	55859	F5902	14	9	330.00357	1.36191	2.02	2.26	2.87	2.75	2.07	QSO	Unknown	Non
114010	J220012.10+013044.7	2011-10-24	55859	F5902	14	10	330.050434	1.51244	2.57	22.44	28.08	45.49	28.6	Star	STAR	K1
114011	J220056.24+012321.9	2011-10-24	55859	F5902	14	11	330.23434	1.38942	1.66	1.92	2.22	2.66	2.14	gal	Unknown	Non
114013	J220024.59+012904.4	2011-10-24	55859	F5902	14	13	330.102468	1.484565	2.65	32.36	49.89	86.31	61.03	Star	STAR	K3
114015	J220058.47+011447.7	2011-10-24	55859	F5902	14	15	330.24365	1.24661	1.38	1.95	1.7	3.23	0.0	gal	Unknown	Non
114023	J220030.68+011939.0	2011-10-24	55859	F5902	14	23	330.12787	1.3275	1.72	2.49	2.64	3.18	2.49	gal	GALAXY	Non
114024	J215936.59+011759.0	2011-10-24	55859	F5902	14	24	329.90247	1.29973	1.57	2.68	3.85	6.22	4.54	gal	Unknown	Non

Figure 17.4: 可以载人其他全天星表,并且叠加

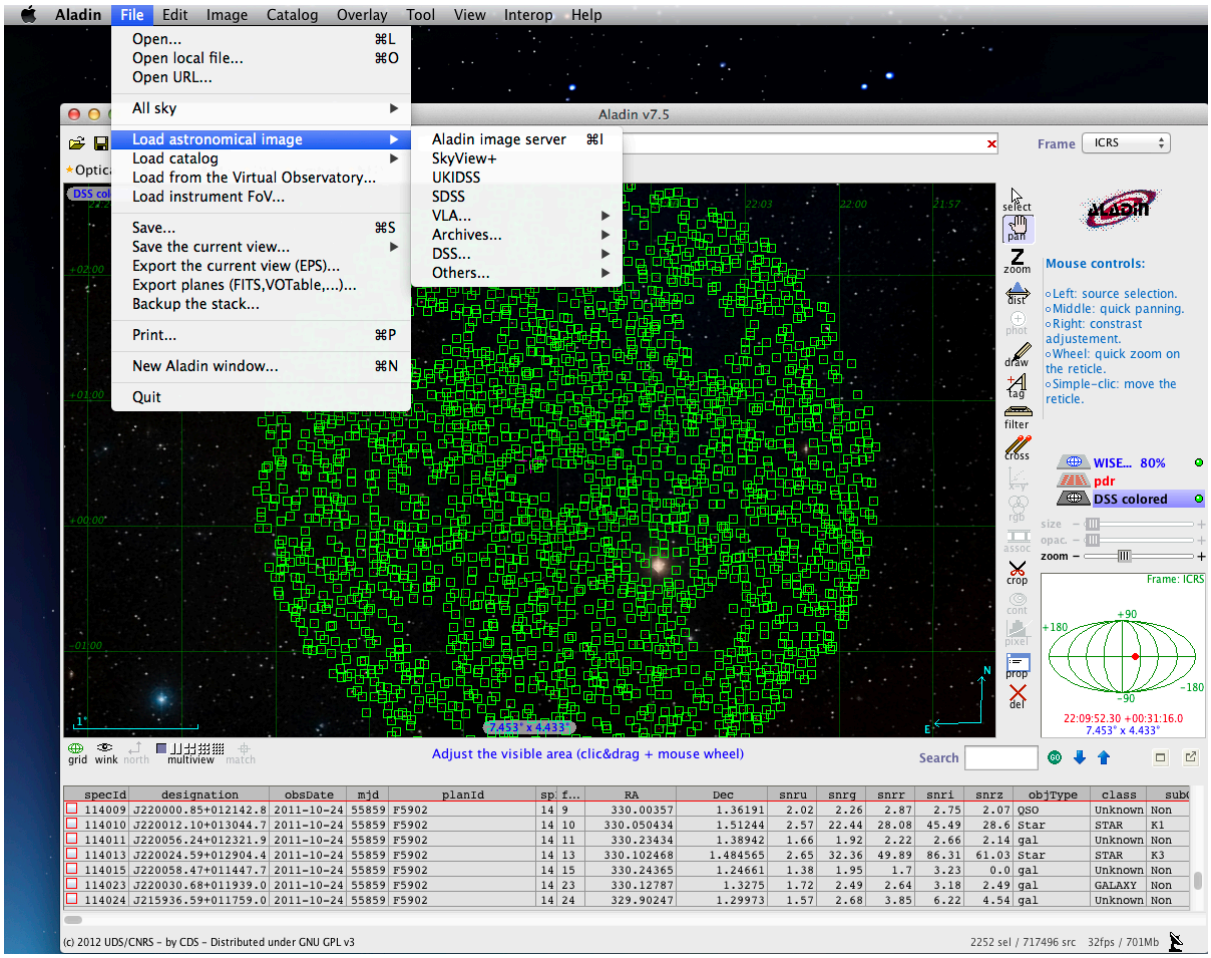


Figure 17.5: 可以载人天文图片

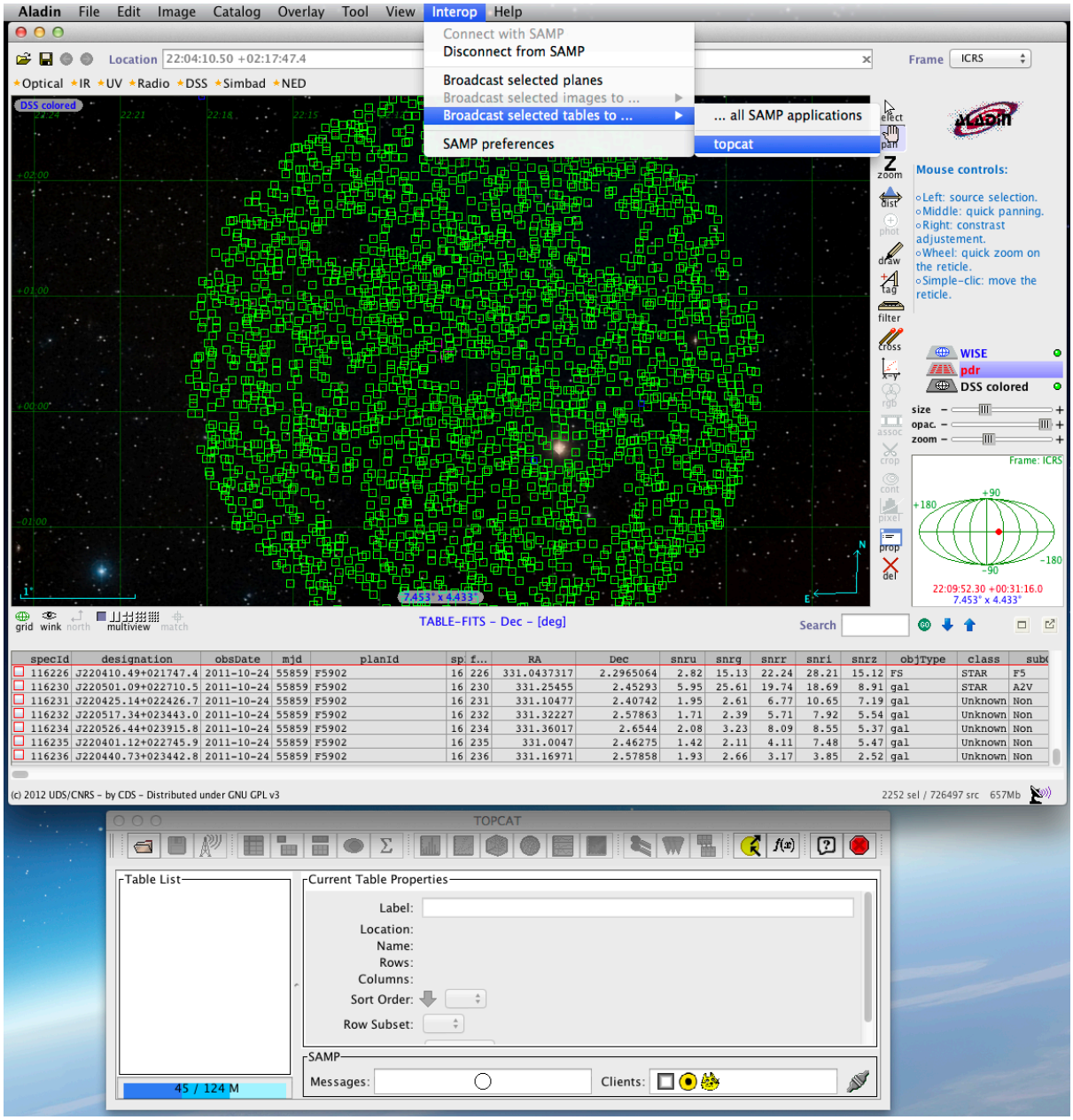


Figure 17.6: SAMP 交互功能, 发送数据到 Topcat

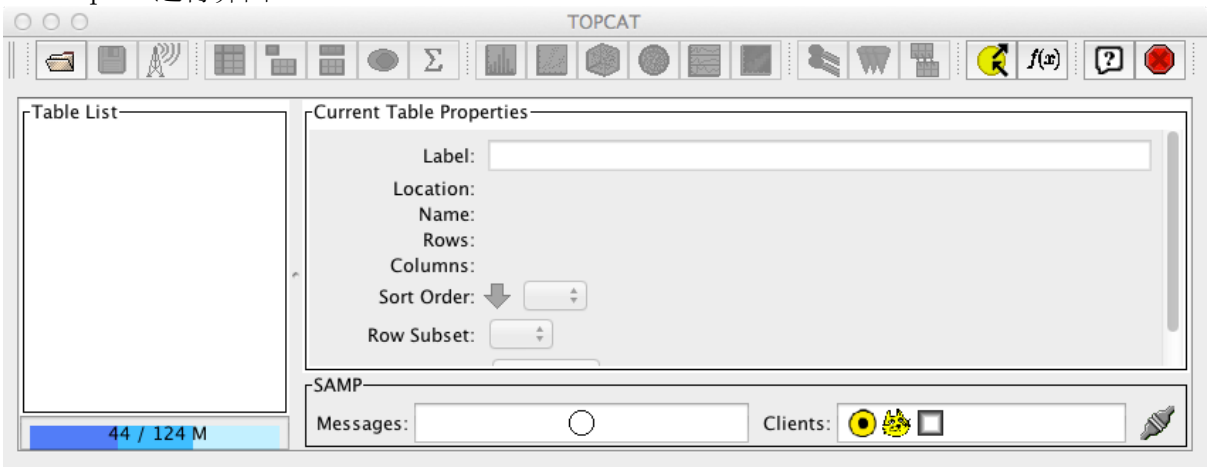
CHAPTER 18

Topcat

星表和表格操作工具或 TOPCAT 是一款表格数据图形阅读器。它提供多种方式来操作数据表格，包括单元数据浏览、有关表格和列的元数据信息显示、数据集的可视化，甚至数据分析等功能。在这一章中，我们讨论 TOPCAT 众多功能中的一小部分。

TOPCAT 最初是作为英国 Starlink 计划的一部分而开发的。现在它由 AstroGrid 进行维护。这个程序由纯 Java 语言编写，遵守 GNU 通用公共许可证（General Public License）。这个软件可以自由下载使用。

Topcat 运行界面



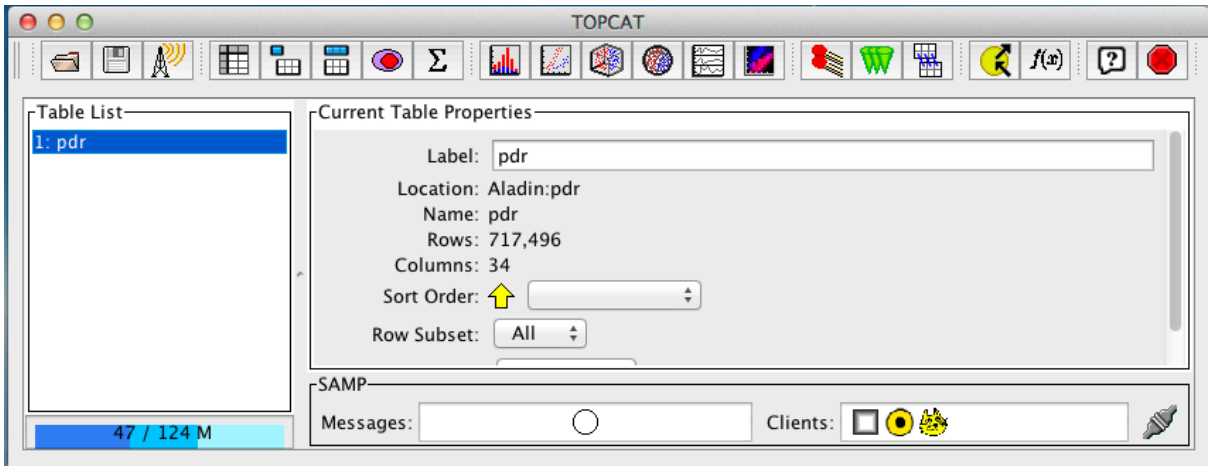
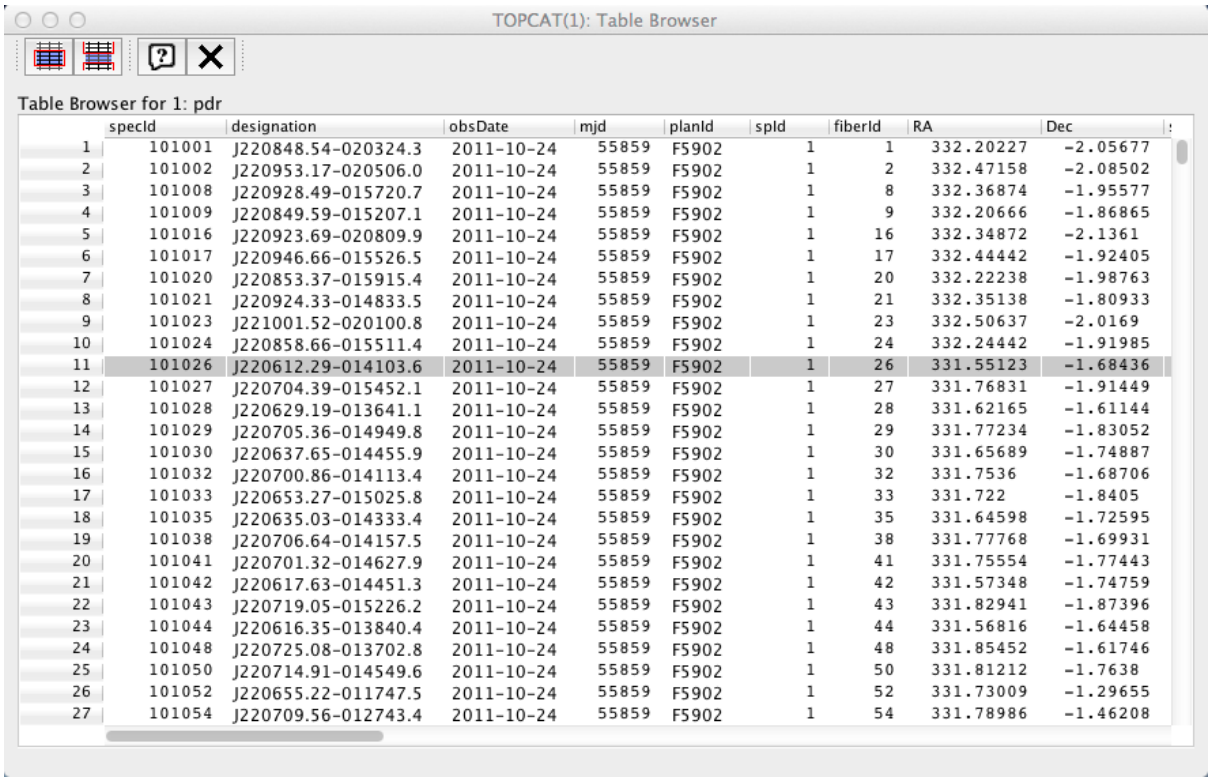
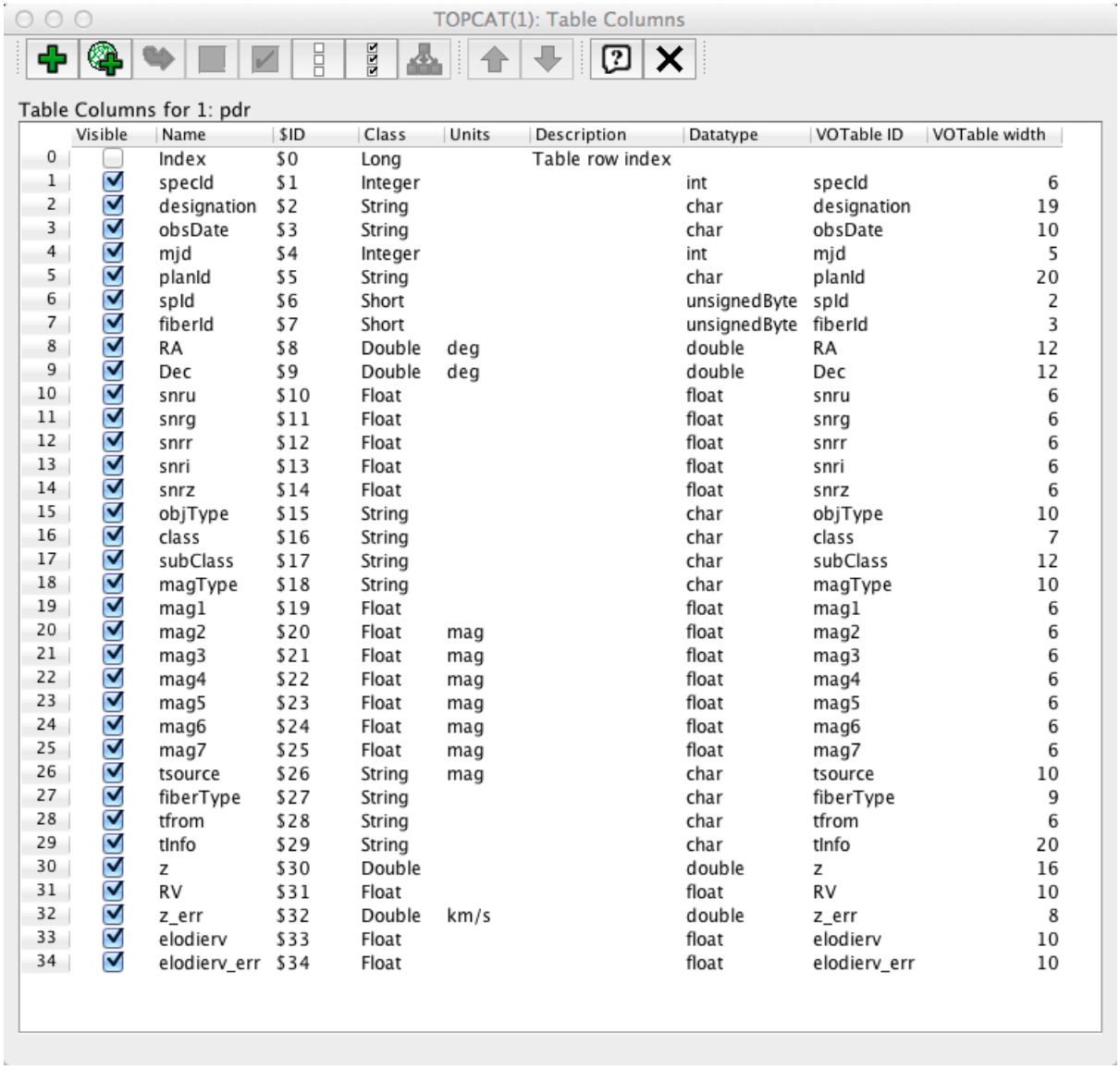


Figure 18.1: 通过 SAMP 协议接受来自 Aladin 的数据



	specId	designation	obsDate	mjd	planId	spId	fiberId	RA	Dec
1	101001	J220848.54-020324.3	2011-10-24	55859	F5902	1	1	332.20227	-2.05677
2	101002	J220953.17-020506.0	2011-10-24	55859	F5902	1	2	332.47158	-2.08502
3	101008	J220928.49-015720.7	2011-10-24	55859	F5902	1	8	332.36874	-1.95577
4	101009	J220849.59-015207.1	2011-10-24	55859	F5902	1	9	332.20666	-1.86865
5	101016	J220923.69-020809.9	2011-10-24	55859	F5902	1	16	332.34872	-2.1361
6	101017	J220946.66-015526.5	2011-10-24	55859	F5902	1	17	332.44442	-1.92405
7	101020	J220853.37-015915.4	2011-10-24	55859	F5902	1	20	332.22238	-1.98763
8	101021	J220924.33-014833.5	2011-10-24	55859	F5902	1	21	332.35138	-1.80933
9	101023	J221001.52-020100.8	2011-10-24	55859	F5902	1	23	332.50637	-2.0169
10	101024	J220858.66-015511.4	2011-10-24	55859	F5902	1	24	332.24442	-1.91985
11	101026	J220612.29-014103.6	2011-10-24	55859	F5902	1	26	331.55123	-1.68436
12	101027	J220704.39-015452.1	2011-10-24	55859	F5902	1	27	331.76831	-1.91449
13	101028	J220629.19-013641.1	2011-10-24	55859	F5902	1	28	331.62165	-1.61144
14	101029	J220705.36-014949.8	2011-10-24	55859	F5902	1	29	331.77234	-1.83052
15	101030	J220637.65-014455.9	2011-10-24	55859	F5902	1	30	331.65689	-1.74887
16	101032	J220700.86-014113.4	2011-10-24	55859	F5902	1	32	331.7536	-1.68706
17	101033	J220653.27-015025.8	2011-10-24	55859	F5902	1	33	331.722	-1.8405
18	101035	J220635.03-014333.4	2011-10-24	55859	F5902	1	35	331.64598	-1.72595
19	101038	J220706.64-014157.5	2011-10-24	55859	F5902	1	38	331.77768	-1.69931
20	101041	J220701.32-014627.9	2011-10-24	55859	F5902	1	41	331.75554	-1.77443
21	101042	J220617.63-014451.3	2011-10-24	55859	F5902	1	42	331.57348	-1.74759
22	101043	J220719.05-015226.2	2011-10-24	55859	F5902	1	43	331.82941	-1.87396
23	101044	J220616.35-013840.4	2011-10-24	55859	F5902	1	44	331.56816	-1.64458
24	101048	J220725.08-013702.8	2011-10-24	55859	F5902	1	48	331.85452	-1.61746
25	101050	J220714.91-014549.6	2011-10-24	55859	F5902	1	50	331.81212	-1.7638
26	101052	J220655.22-011747.5	2011-10-24	55859	F5902	1	52	331.73009	-1.29655
27	101054	J220709.56-012743.4	2011-10-24	55859	F5902	1	54	331.78986	-1.46208

Figure 18.2: 查看数据文件



	Visible	Name	$ID	Class	Units	Description	Datatype	VOTable ID	VOTable width
0	☐	Index	$0	Long		Table row index			
1	☑	specId	$1	Integer			int	specId	6
2	☑	designation	$2	String			char	designation	19
3	☑	obsDate	$3	String			char	obsDate	10
4	☑	mjd	$4	Integer			int	mjd	5
5	☑	planId	$5	String			char	planId	20
6	☑	spId	$6	Short			unsignedByte	spId	2
7	☑	fiberId	$7	Short			unsignedByte	fiberId	3
8	☑	RA	$8	Double	deg		double	RA	12
9	☑	Dec	$9	Double	deg		double	Dec	12
10	☑	snru	$10	Float			float	snru	6
11	☑	snrg	$11	Float			float	snrg	6
12	☑	snrr	$12	Float			float	snrr	6
13	☑	snri	$13	Float			float	snri	6
14	☑	snrz	$14	Float			float	snrz	6
15	☑	objType	$15	String			char	objType	10
16	☑	class	$16	String			char	class	7
17	☑	subClass	$17	String			char	subClass	12
18	☑	magType	$18	String			char	magType	10
19	☑	mag1	$19	Float			float	mag1	6
20	☑	mag2	$20	Float	mag		float	mag2	6
21	☑	mag3	$21	Float	mag		float	mag3	6
22	☑	mag4	$22	Float	mag		float	mag4	6
23	☑	mag5	$23	Float	mag		float	mag5	6
24	☑	mag6	$24	Float	mag		float	mag6	6
25	☑	mag7	$25	Float	mag		float	mag7	6
26	☑	tsource	$26	String	mag		char	tsource	10
27	☑	fiberType	$27	String			char	fiberType	9
28	☑	tfrom	$28	String			char	tfrom	6
29	☑	tInfo	$29	String			char	tInfo	20
30	☑	z	$30	Double			double	z	16
31	☑	RV	$31	Float			float	RV	10
32	☑	z_err	$32	Double	km/s		double	z_err	8
33	☑	elodierv	$33	Float			float	elodierv	10
34	☑	elodierv_err	$34	Float			float	elodierv_err	10

Figure 18.3: 查看数据列信息

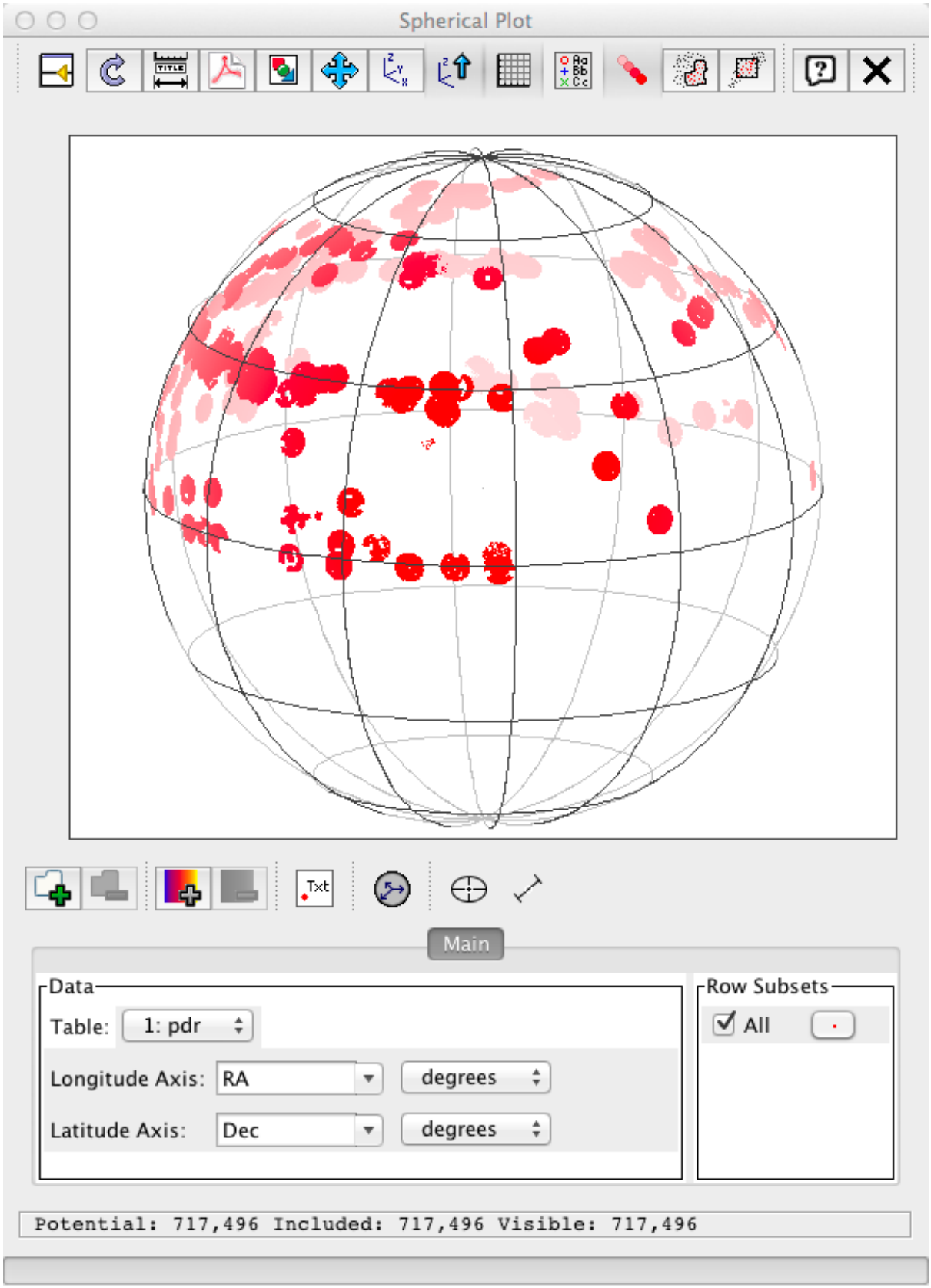


Figure 18.4: 星表球面可视化

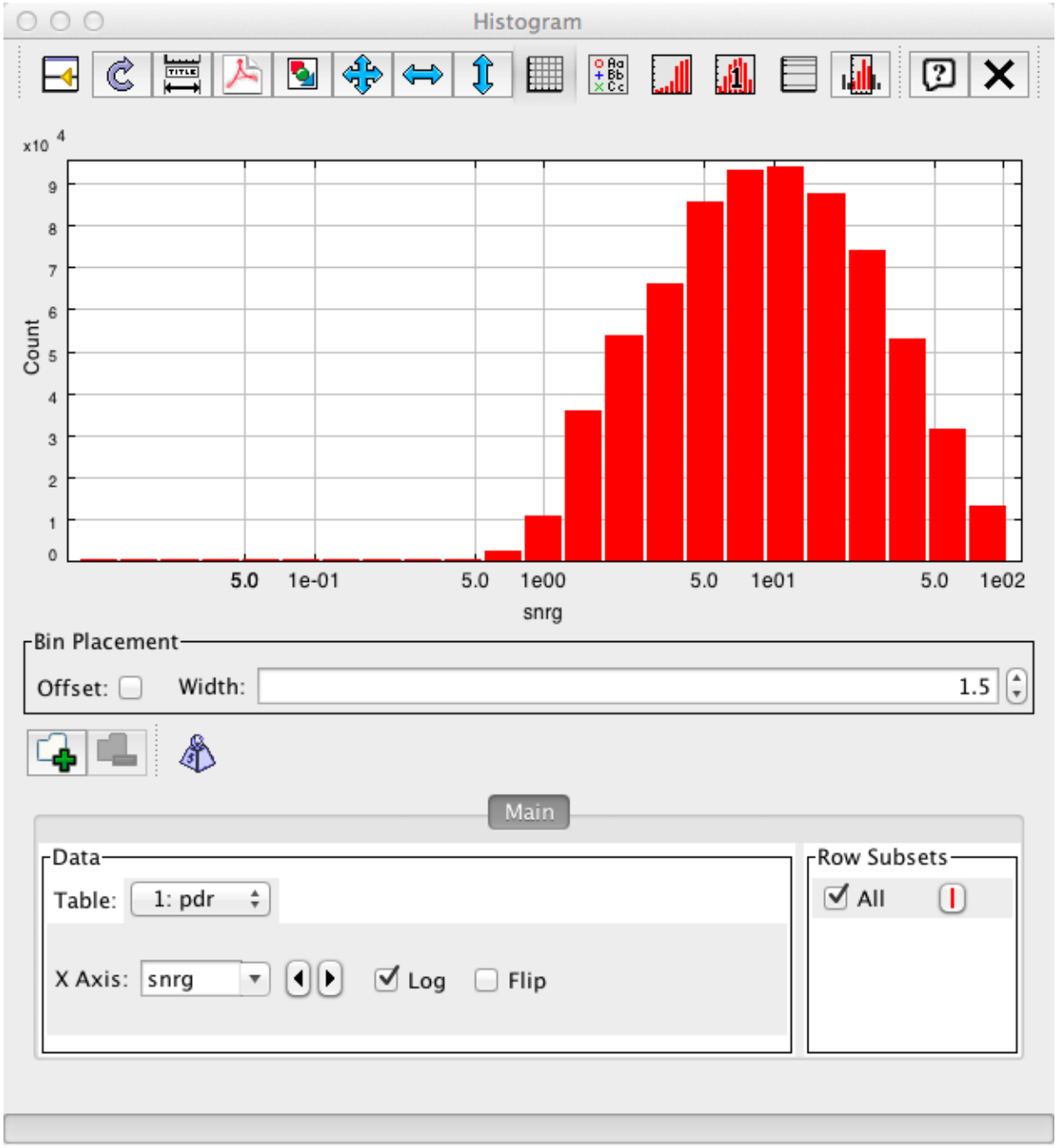
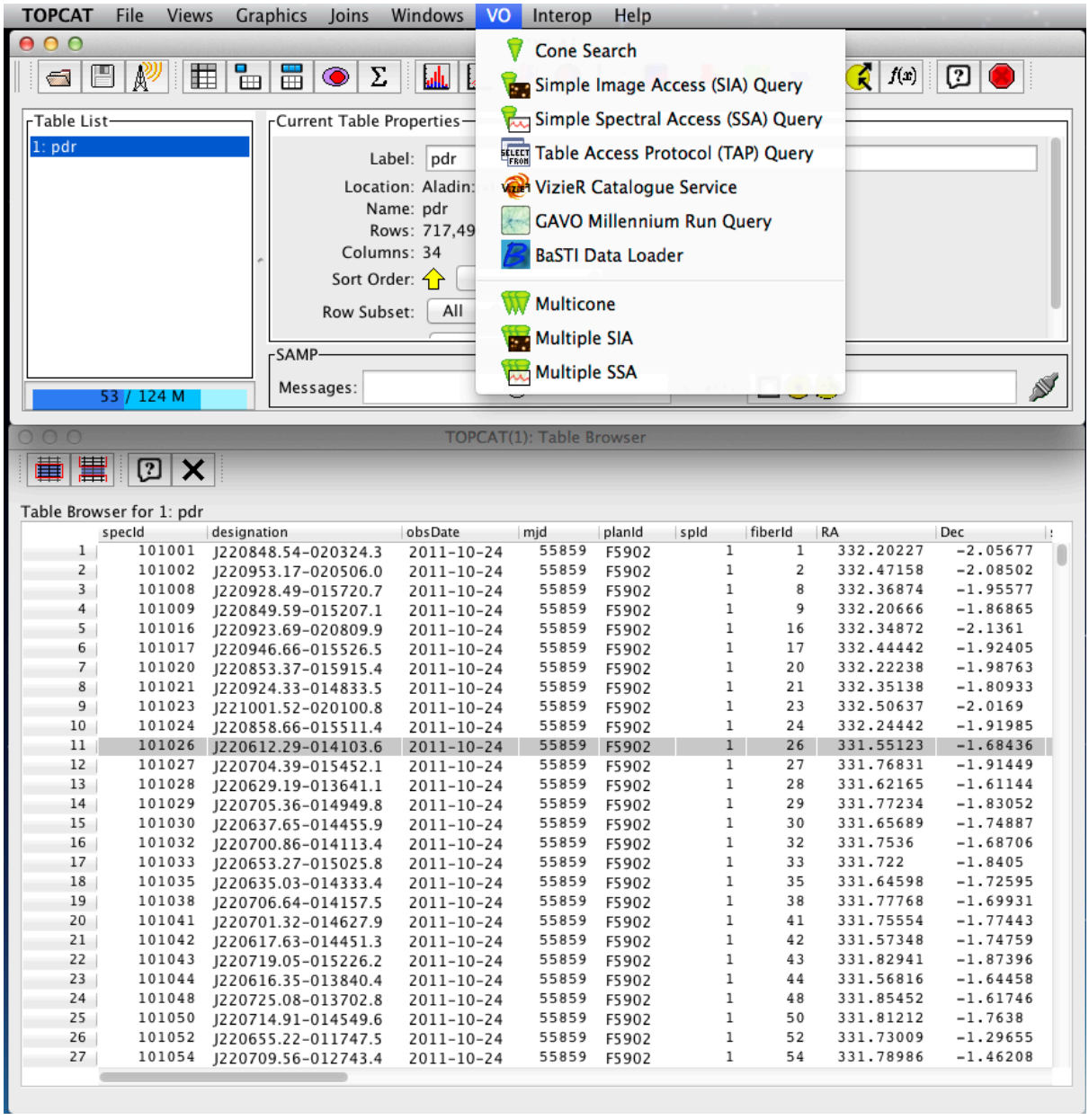


Figure 18.5: 数据分析



	specId	designation	obsDate	mjd	planId	spId	fiberId	RA	Dec
1	101001	J220848.54-020324.3	2011-10-24	55859	F5902	1	1	332.20227	-2.05677
2	101002	J220953.17-020506.0	2011-10-24	55859	F5902	1	2	332.47158	-2.08502
3	101008	J220928.49-015720.7	2011-10-24	55859	F5902	1	8	332.36874	-1.95577
4	101009	J220849.59-015207.1	2011-10-24	55859	F5902	1	9	332.20666	-1.86865
5	101016	J220923.69-020809.9	2011-10-24	55859	F5902	1	16	332.34872	-2.1361
6	101017	J220946.66-015526.5	2011-10-24	55859	F5902	1	17	332.44442	-1.92405
7	101020	J220853.37-015915.4	2011-10-24	55859	F5902	1	20	332.22238	-1.98763
8	101021	J220924.33-014833.5	2011-10-24	55859	F5902	1	21	332.35138	-1.80933
9	101023	J221001.52-020100.8	2011-10-24	55859	F5902	1	23	332.50637	-2.0169
10	101024	J220858.66-015511.4	2011-10-24	55859	F5902	1	24	332.24442	-1.91985
11	101026	J220612.29-014103.6	2011-10-24	55859	F5902	1	26	331.55123	-1.68436
12	101027	J220704.39-015452.1	2011-10-24	55859	F5902	1	27	331.76831	-1.91449
13	101028	J220629.19-013641.1	2011-10-24	55859	F5902	1	28	331.62165	-1.61144
14	101029	J220705.36-014949.8	2011-10-24	55859	F5902	1	29	331.77234	-1.83052
15	101030	J220637.65-014455.9	2011-10-24	55859	F5902	1	30	331.65689	-1.74887
16	101032	J220700.86-014113.4	2011-10-24	55859	F5902	1	32	331.7536	-1.68706
17	101033	J220653.27-015025.8	2011-10-24	55859	F5902	1	33	331.722	-1.8405
18	101035	J220635.03-014333.4	2011-10-24	55859	F5902	1	35	331.64598	-1.72595
19	101038	J220706.64-014157.5	2011-10-24	55859	F5902	1	38	331.77768	-1.69931
20	101041	J220701.32-014627.9	2011-10-24	55859	F5902	1	41	331.75554	-1.77443
21	101042	J220617.63-014451.3	2011-10-24	55859	F5902	1	42	331.57348	-1.74759
22	101043	J220719.05-015226.2	2011-10-24	55859	F5902	1	43	331.82941	-1.87396
23	101044	J220616.35-013840.4	2011-10-24	55859	F5902	1	44	331.56816	-1.64458
24	101048	J220725.08-013702.8	2011-10-24	55859	F5902	1	48	331.85452	-1.61746
25	101050	J220714.91-014549.6	2011-10-24	55859	F5902	1	50	331.81212	-1.7638
26	101052	J220655.22-011747.5	2011-10-24	55859	F5902	1	52	331.73009	-1.29655
27	101054	J220709.56-012743.4	2011-10-24	55859	F5902	1	54	331.78986	-1.46208

Figure 18.6: 访问在线的 VO 服务

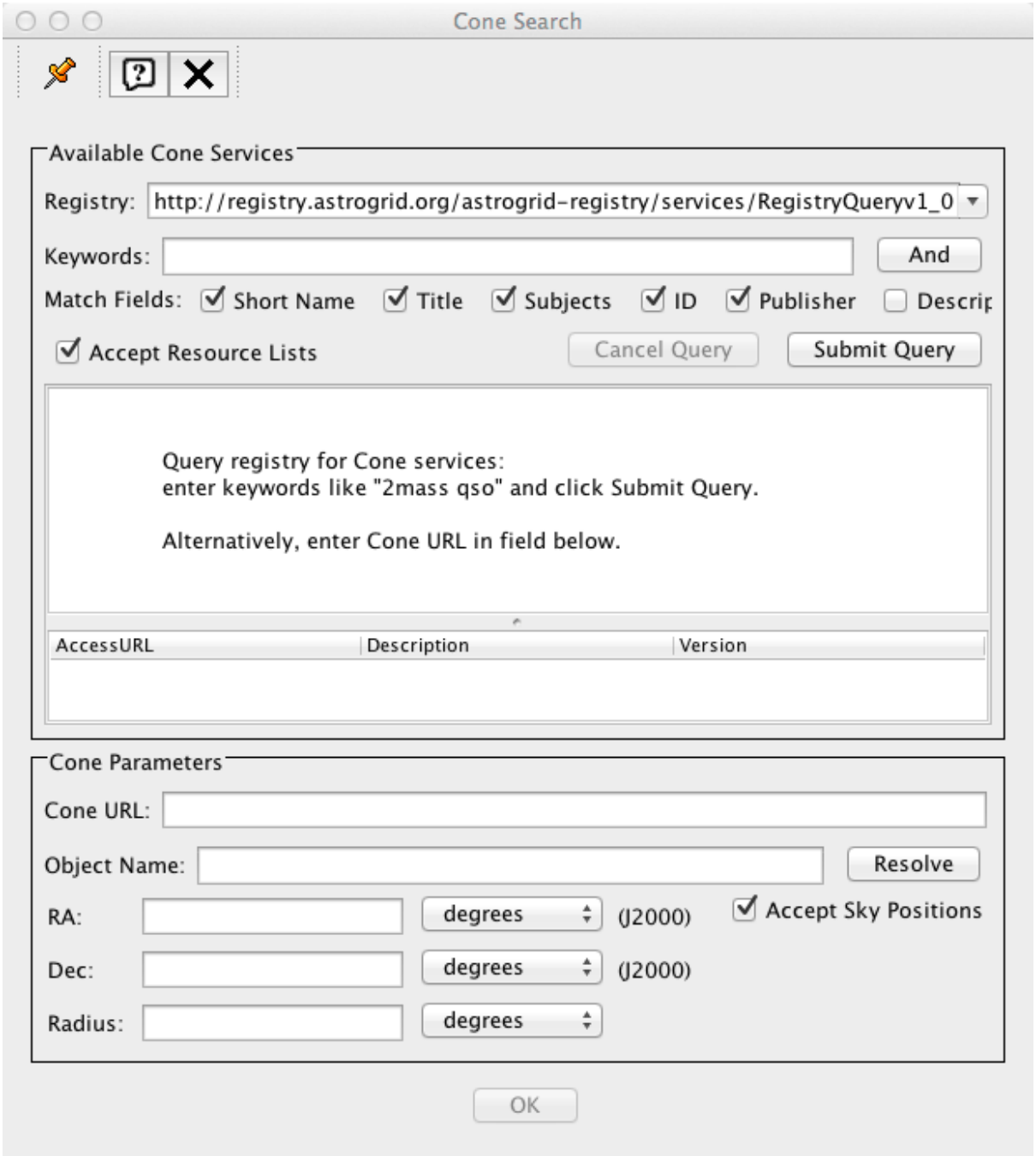


Figure 18.7: 访问在线的 VO 服务

Bibliography

[1] Arnold Robbins, Nelson H. F. Beebe, O'Reilly Taiwan 公司编译 `Shell` 脚本学习指南,2009: 机械工业出版社, 北京

[2] Wes McKinney, 唐学韬等译 利用 `Python` 进行数据分析,2013: 机械工业出版社, 北京

[3] Astropy, http://docs.astropy.org/en/stable/

[4] Python Data Analysis Library, http://pandas.pydata.org/

[5] Ivan Idris, 张崇明译 `NumPy` 攻略,2013: 人民邮电出版社, 北京

[6] Abraham Silberschatz 等, 杨冬青等译 数据库系统概念,2006: 机械工业出版社, 北京