

Build Your Own SkyNode!

Norbert Purger¹, Tamás Budavári², Alexander S. Szalay², Ani Thakar²
and István Csabai¹

Abstract. SkyQuery is an excellent VO prototype application that marries Web Services technology with emerging VO standards to enable dynamic cross-matching queries between different VO-enabled archives. The archive data is stored in databases that are published online as SkyNodes.

As the available data from Sky Surveys and new digital archives rapidly multiplies every year, more than 80 percent of the data will exist outside of large data centers at any given moment, making it very important to have dynamic cross-identification tools like SkyQuery.

Loading an entire survey like 2MASS or SDSS into a database involves making decisions about issues like data formats and indices for tables. We describe the process of loading such a large amount of data into a relational DBMS (SQL Server) and generating a sky index using the Hierarchical Triangular Mesh (HTM), which provides a really fast way to find objects. This can be easily done even for a large survey like the 2MASS All-Sky Data Release (150GB uncompressed, 471M objects) in as little as 2 days including the required computation time for HTM.

1. Building a SkyNode

1.1. Preparing The Data

Sky Surveys usually provide data to the public in a form of ASCII or FITS files and their brief descriptions. Using these descriptions we are able to understand the structure of the files and the data types included in them. If we would like to load this data in a database, we'll have to create a representation of the survey data types suitable for our database system. This means a database schema and a parser to convert the files to our format.

Most of the time we don't need to deal with lot's of different tables, it is enough to make only one or two tables for all of our data (see an example of a more complex schema in the next section, Fig. 2). This table (PhotoObj or SpecObj) will hold all the data we can find in the public files (but with SQL data types) completed with an identity primary key (ObjID) column. The best way to make this "create table" script is to write a description after every column as

¹Dept. of Physics of Complex Systems, Eötvös Loránd University, H-1117 Budapest, Hungary

²Dept. of Physics & Astronomy, Johns Hopkins University, Baltimore, MD 21218, USA

a comment. By doing this, we'll be able to generate important metadata to our tables and functions later on.

Our next step is the file conversion. In order to load our files into a database, the clearest way is to convert them into a character delimited (CSV) format using the new data types and our symbol of a NULL value. Additionally, we must include our coordinates in a float type J2000 format as well (ra, dec), because the HTM coordinates will be calculated from these values. If we couldn't find any sample code bundled with the public data, we can still start thinking about using regular expressions with our parser.

Name	Type	Length	Nullable	description
objID	bigint	8	no	unique object identifier, SEQNUM in original catalog
cat	char	1	no	Catalogue Type: n for NGP, s for SGP and r for random fields
spectra	smallint	2	no	Number of spectra obtained
name	varchar	10	no	2dFGRS name
UKST	varchar	3	no	UKST plate (=FIELD)
ra	float	8	no	Right Ascension (J2000)
dec	float	8	no	Declination (J2000)
bjg	real	4	no	Final bj magnitude without extinction correction
bjsel	real	4	no	Final bj magnitude with extinction correction
bjg_old	real	4	no	Original bj magnitude without extinction correction
bjselold	real	4	no	Original bj magnitude with extinction correction
galext	real	4	no	Galactic extinction value
sb_bj	real	4	no	SuperCosmos bj magnitude without extinction correction
sr_r	real	4	no	SuperCosmos R magnitude without extinction correction
z	real	4	no	Best redshift (observed)
z_helio	real	4	no	Best redshift (heliocentric)
obsrun	varchar	5	no	Observation run of best spectrum
quality	smallint	2	no	Redshift quality parameter for best spectrum; reliable redshifts have >=3
abemma	smallint	2	no	Redshift type (abs=1, emi=2, man=3)
z_abs	real	4	no	Cross-correlation redshift from best spectrum
kbestr	smallint	2	no	Cross-correlation template from best spectrum
r_crcor	real	4	no	Cross-correlation R value from best spectrum
z_emi	real	4	no	Emission redshift from best spectrum
nmbest	smallint	2	no	Number of emission lines for Z_EMI from best spectrum
snr	real	4	no	Median S/N per pixel from best spectrum
eta_type	real	4	no	Eta spectral type parameter from best spectrum (-99.9 if none)

Figure 1. Metadata of the 2dFGRS SkyNode at SkyQuery.net

1.2. Creating The Database

SkyNodes are using HTM2 coordinates to index (htmID) and locate survey objects. The HTM2 code that will calculate an htmID in the database is realized by an extended stored procedure, which must be installed on the server. Our next step is to create the database that will hold all of our survey-related data (130%-150% of the uncompressed data size). To avoid huge transactional logs, this database should use Simple or Bulk-logged recovery model. By now we are ready to create the structure of our new SkyNode by executing our “create table” script and the ones that are required by every SkyNode (e.g. HTM2, neighbor and matching functions).

At this point we have a fully functional database without a row of data. To fill it up, we can use “bulk load” with the converted CSV files, or advanced (graphical) tools (e.g. Data Transformation Services in SQL Server). After this, the HTM coordinates will be calculated and loaded into a separate table by using a function of the installed extended stored procedure.

To speed up the searches in our node, we should create additional indices and define relationships between our tables (e.g. between PhotoObj and HTM tables using ObjID)

1.3. Publishing The Node

If we would like to work with data from several different surveys, it is really necessary to have some kind of description about the actual data we are using. If we write comments to every new object (tables, columns, functions) in the SQL scripts, we are able to generate and load this important metadata into our database (Fig. 1).

In order to make a SkyNode, our last two steps are creating a user to access the data and set up the web service using a modified configuration file. Our user must have permissions to SELECT from the tables, and EXEC the stored procedures. The SkyNode configuration file must include the user name, password and a few lines of description (name, location, ..).

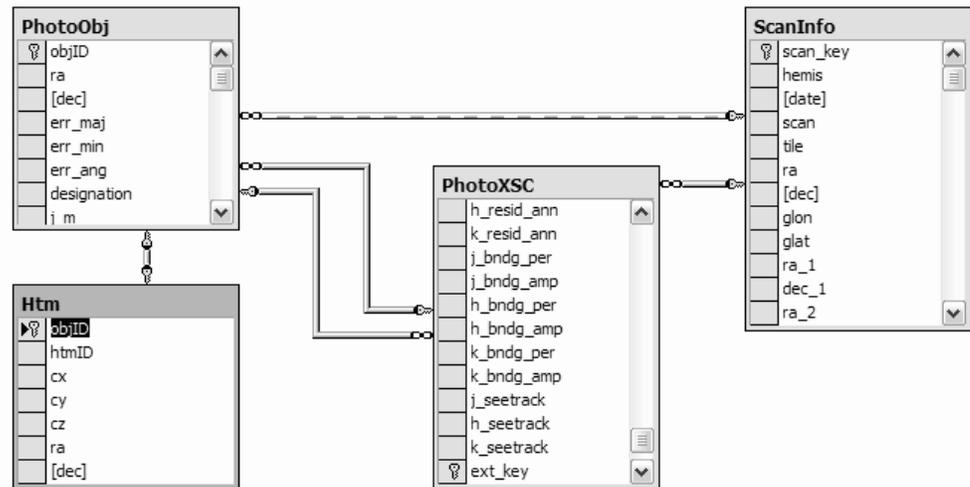


Figure 2. Database schema for the 2MASS All-Sky Data Release

2. The 2MASS SkyNode

With its more than 471 million objects the 2MASS All-Sky Data Release is a bit more challenging to handle. The Data Release includes 3 different type of “catalog” (Point Source Catalog, Extended Source Catalog, ScanInfo), which makes the database schema more complex (Fig. 2). Even if we have a more

complex database, the biggest problem is still the time required to load the data and calculate the coordinate based indices. Using the new HTM2 code to create the htmID indices came up to be a surprisingly fast solution. After about 36 hours of parallel loading and computing time on 2 dual Xeon processors, we had a 2MASS database in the SQL Server.

3. New SkyNodes at SkyQuery.net

New, final data releases from great surveys made a good choice to extend the list of SkyNodes this year. After about one month of work, the following new nodes were set up at SkyQuery.net:

- FIRST (2003 April 11 release)
- IRAS (point source catalog)
- 2dFGRS (APM Source and Spectra data)
- 2QZ (full catalog)
- NVSS (full catalog)
- PSCz (full catalog)
- 2MASS (All-Sky Data R., Point and Extended Source Catalog, ScanInfo)

To find more information about SkyNodes and SkyQuery, or to download prepared SQL scripts visit the following pages:

<http://www.skyquery.net>

<http://skyserver.elte.hu/skynode>

Acknowledgments. This work was supported by the National Science Foundation's Information Technology Research Program under Cooperative Agreement AST0122449 with The Johns Hopkins University and by the Hungarian Scientific Research Fund OTKA T037548.

References

- Budavári, T., et al. 2003, in ASP Conf. Ser., Vol. 295, ADASS XII, ed. H. E. Payne, R. I. Jedrzejewski, & R. N. Hook (San Francisco: ASP), 31
- Fekete, Gy., et al. 2004, this volume, 289
- Malik, T., et al. 2002, CIDR'03, p.17, '*SkyQuery: A Webservice Approach to Federate Databases*'
- Thakar, A., et al. 2004, this volume, 38