

The SDSS SkyServer – Public Access to the Sloan Digital Sky Server Data

Alexander S. Szalay¹, Jim Gray², Ani R. Thakar¹, Peter Z. Kunszt⁴, Tanu Malik¹,
Jordan Raddick¹, Christopher Stoughton³, Jan vandenBerg¹

(1) The Johns Hopkins University, (2) Microsoft, (3) Fermi National Accelerator Laboratory, Batavia, (4) CERN
Gray@Microsoft.com, {Szalay, Thakar, Raddick, Vincent}@pha.jhu.edu, Peter.Kunszt@cern.ch, Stoughto@fnal.gov

ABSTRACT

The SkyServer provides Internet access to the public Sloan Digital Sky Survey (SDSS) data for both astronomers and for science education. This paper describes the SkyServer goals and architecture. It also describes our experience operating the SkyServer on the Internet. The SDSS data is public and well-documented so it makes a good test platform for research on database algorithms and performance.

1. Introduction

The SkyServer provides Internet access to the public Sloan Digital Sky Survey (SDSS) data for both astronomers and for science education. The SDSS is a 5-year survey of the Northern sky (10,000 square degrees) to about 1/2 arcsecond resolution using a modern ground-based telescope [SDSS]. It will characterize about 200M objects in 5 optical bands, and will measure the spectra of a million objects. The first year's data is now public.

The raw data gathered by the SDSS telescope at Apache Point, New Mexico, is processed by software data analysis pipelines at Fermilab. Imaging pipelines analyze data from the camera to extract about 400 attributes for each celestial object along with a 5-color "cutout" image. The spectroscopic pipelines analyze data from the spectrographs, to extract calibrated spectra, redshifts, absorption and emission lines, and many other attributes. These pipelines embody much of mankind's knowledge of astronomy [SDSS-EDR]. The pipeline software is a major part of the SDSS project: approximately 25% of the project's cost and effort. The result is a high-quality catalog of the Northern sky, and of a small stripe of the Southern sky. When complete, the survey data will occupy about 25 terabytes (TB) of source data, and about 13 TB of processed data, for a total of nearly 40 TB.

After calibration, the pipeline output is available to the SDSS consortium astronomers. After approximately a year, the SDSS publishes the data to the astronomy community and the public – so in 2007 all the data will be available to everyone everywhere.

The first year's SDSS data is now public. It is 80GB containing about 14 million objects and 50 thousand spectra. You can access it via the SkyServer (<http://skyserver.sdss.org/>) or you may get a private copy of the data. The web server supports both professional astronomers and educational access.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACM SIGMOD' 2002, June 4-6, 2002, Madison, Wisconsin, USA
Copyright 2002 ACM 1-58113-497-5/02/06

Amendments to the public SDSS data will be released as the data analysis pipeline improves, and the data will be augmented as more becomes public (next scheduled release is January 2003). In addition, the SkyServer will get better documentation and tools as we learn how it is used. There are Japanese and German versions of the website, and the server is being mirrored in many parts of the world.

This paper sketches the SkyServer database and web site design, describes the data loading pipeline, and reports on website usage.

2. Web Server Interface Design

The SkyServer is accessed via the Internet using standard browsers. It accepts point-and-click requests for images of the sky, images of spectra, and for tabular outputs of the SDSS database. It also has links to the online literature about objects (e.g. NED, VizieR and Simbad). The site has an SDSS project description, tutorials on how the data was collected and what it means, and also has projects suitable to teach or learn astronomy and computational science at various grade levels. Figure 1 cartoons the main access screens.

The simplest and most popular access is a coffee-table atlas of *famous places* that shows color images of interesting (and often famous) celestial objects. These images try to lead the viewer to articles about these objects, and let them drill down to view the objects within the SDSS data. There are also tools that let the user to get images and spectra of particular objects (see Figure 1). To drill down further, there is a text and a GUI SQL interface that lets sophisticated users mine the SDSS database. A point-and-click pan-zoom scheme lets users pan across a section of the sky and pick objects and their spectra (if present).

The sky color images were built specially for the website. The original 5-color 80-bit deep images were converted using a nonlinear intensity mapping to reduce the brightness dynamic range to screen quality. The augmented-color images are 24bit RGB, stored as JPEGs. An image pyramid was built at 4 zoom levels. The spectra are also converted to 8bit GIF images.

The Alfred P. Sloan Foundation, the Participating Institutions, the National Aeronautics and Space Administration, the National Science Foundation, the U.S. Department of Energy, the Japanese Monbukagakusho, and the Max Planck Society have provided funding for the creation and distribution of the SDSS Archive. The SDSS Web site is <http://www.sdss.org/>. The Participating Institutions are University of Chicago, Fermilab, Institute for Advanced Study, Japan Participation Group, Johns Hopkins University, Max-Planck-Institute for Astronomy (MPIA), Max-Planck-Institute for Astrophysics (MPA), New Mexico State University, Princeton University, United States Naval Observatory, and University of Washington. Compaq donated the hardware for the SkyServer and some other SDSS processing. Microsoft donated the basic software for the SkyServer.

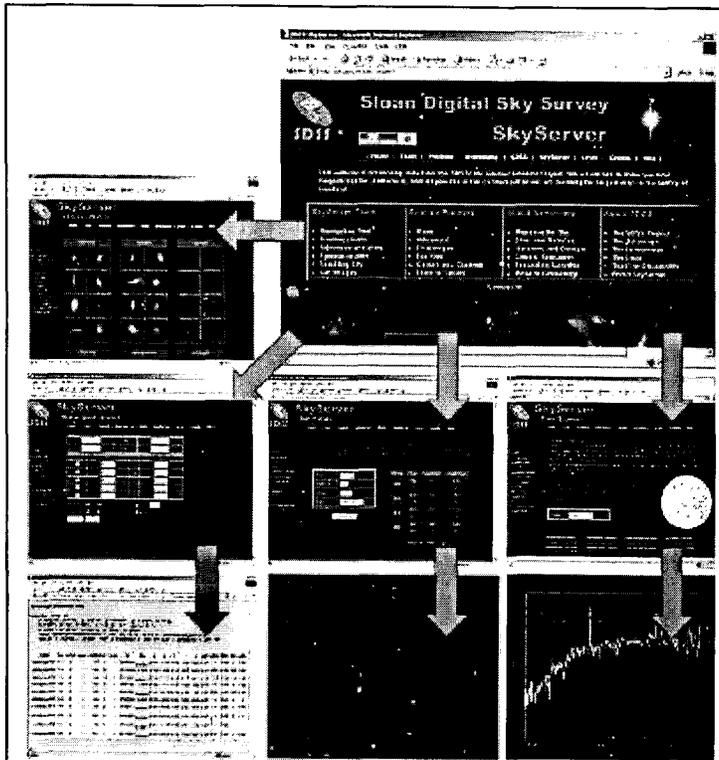


Figure 1: The SkyServer web interface provides many different ways to look at the SDSS data. The simplest is “famous places” which is just a gallery of beautiful images. More sophisticated users can navigate to find the images and the data for a particular celestial object. There are a variety of query interfaces and also links to the online literature about objects.

The SkyServer is just one of the ways to access the SDSS data. There is also the Catalog Archive Server (CAS) which is an ObjectivityDB™ database built by Johns Hopkins University (<http://www.sdss.jhu.edu/ScienceArchive/>). Much of the SkyServer database architecture is copied from the CAS database

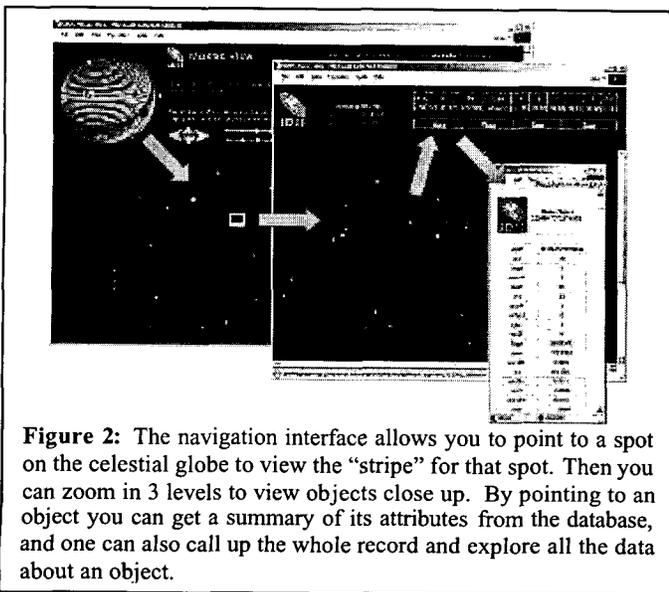


Figure 2: The navigation interface allows you to point to a spot on the celestial globe to view the “stripe” for that spot. Then you can zoom in 3 levels to view objects close up. By pointing to an object you can get a summary of its attributes from the database, and one can also call up the whole record and explore all the data about an object.

design to leverage its documentation. In addition, the raw SDSS pixel-level files are available from Data Archive Server (DAS) at Fermilab (<http://sdssdp7.fnal.gov/cgi-bin/das/main.cgi>). The CAS and DAS are operated by Fermilab and accessed via Space Telescope Science Institute’s MAST (Multi Mission Archive at Space Telescope) website at <http://archive.stsci.edu/sdss/>

3. SkyServer Data Mining

Data mining was our original motive to build the SQL-based SkyServer. We wanted a tool that would be able to quickly answer questions like: “find gravitational lens candidates” or “find other objects like this one.” Indeed, we [Szalay] defined 20 typical queries and designed the SkyServer database to answer those queries. Another paper describes the queries and their performance in detail and that paper is summarized in section 11 [Gray].

The queries correspond to typical tasks astronomers would do with a C++ program, extracting data from the archive, and then analyzing it. Being able to state queries simply and quickly could be a real productivity gain for the Astronomy community. We were surprised and pleased to discover that all 20 queries have fairly simple SQL equivalents. Often the query can be expressed as a single SQL statement. In some cases, the query is iterative, the results of one query feeds into the next.

Many of the queries run in a few seconds. Some involving a sequential scan of the database take about 3 minutes. A few complex joins take nearly an hour. Occasionally the SQL optimizer picks a poor plan and a query can take several hours – though this did not happen on the 20 queries. The spatial data queries are both simple to state and execute quickly using a spatial index. We circumvented a limitation in SQL Server by pre-computing the neighbors of each object. Even without being forced to do it, we might have created this materialized view to speed queries. In general, the queries benefited from indices and column subsets containing popular fields.

Translating the queries into SQL requires a good understanding of astronomy, a good understanding of SQL, and a good understanding of the database. “Normal” astronomers use very simple SQL queries. They use SQL to extract a subset of the data and then analyze that data on their own system using their own tools. SQL, especially complex SQL involving joins and spatial queries, is just not part of the current astronomy toolkit. This stands as a barrier to wider use of the SkyServer by the astronomy community. A good visual query tool that makes it easier to compose SQL would ameliorate this problem.

4. SkyServerQA-The SDSS Query Tool

SkyServerQA is a GUI SQL query tool to help compose SQL queries. It was inspired by the SQL Server Query Analyzer, but runs as a Java applet on UNIX, Macintosh, and Windows clients and is freely available from the SDSS web site [Malik]. It connects via ODBC/JDBC (for local use) and via HTTP or SOAP for use over the Internet.

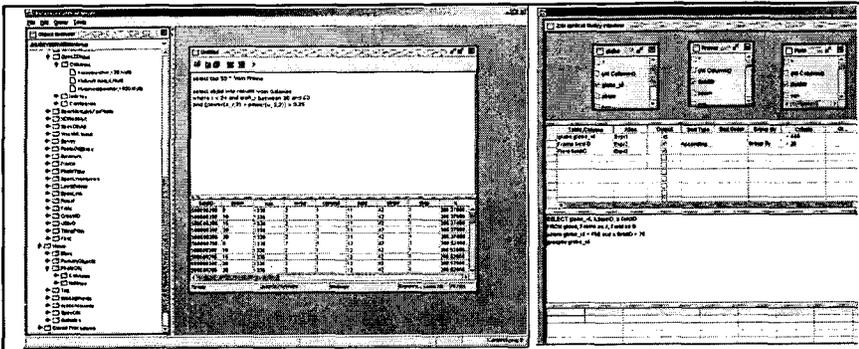


Figure 3: The SkyServerQA is a public domain Java applet that runs on Unix, Macintosh, and Windows clients. It can be used to query the SkyServer database. It has a text and a GUI input mode. The Object Browser (left pane) gets the database schema from the server.

SkyServerQA provides both a text-based and a diagram-based query mode. In the text-based mode, the user composes and executes SQL queries, stored procedures, or functions. The text-based query window is shown on the left of Figure 3. In the diagram-based mode, the user formulates the query from icons, lists, and options in the left pane, without needing to know any syntax. While the user creates the query diagram, SkyServerQA creates the syntactically correct SQL query. This implicitly teaches SQL.

SkyServerQA is a hierarchical object browser of the database, tables, stored procedures, functions, columns, indexes, dependencies, and comments (see left pane of Figure 3). When a table or field is selected a tool tip popup gives a brief text description of the object. Metadata includes data types, lengths, and null indicators. Indices consist of the columns on which they are built. Constraints show the Primary Key constraint for the table as well as Foreign Key constraints. Foreign Key constraints show the table to which they reference.

SkyServerQA provides results in three formats

1. Grid Based for quick viewing,
 2. Column Separated Values (CSV) ASCII for use in spreadsheets and text tools,
 3. XML for applications that can read XML data,
 4. FITS is a file format widely used in astronomy [FITS].
- The user can save these results to a file.

Query execution statistics are vital for large result-sets. The status window shows the execution time of each query, rounded

to the nearest second. It also shows the connection information of the user, catalog name and server name.

The public SkyServer limits queries to 1,000 records or 30 seconds of computation. For more demanding queries, the users must use a private SkyServer.

Once the query answer is produced, there is still a need to understand it. We have made no progress on the data visualization problems posed in [Szalay].

5. Web Server Design

The SkyServer's architecture is fairly simple: a front-end IIS web server accepts HTTP

requests processed by JavaScript Active Server Pages (ASP). These scripts use Active Data Objects (ADO) to query the backend SQL database server. SQL returns record sets that the JavaScript formats into pages. The website is about 10,000 lines of JavaScript and was built by two people as a spare-time activity.

This design derives from the TerraServer [Barclay] – both the structured data and the images are all stored in the SQL database. A 4-level image pyramid of the images is precomputed, allowing users to see an overview of the sky, and then zoom into specific areas for a close-up view of a particular object.

The most challenging aspect of web site design is supporting a rich user interface for many different browsers. Supporting Netscape Navigator™, Mozilla™, Opera™, and Microsoft Internet Explorer™ is a challenge – especially when the many Windows™, Macintosh™, and UNIX™ variants are considered. We also support PDA and PocketPC browsers that have limited JavaScript and no Cascading Style Sheet support. The SkyServer does not download applets to the clients (except for SkyServerQA), but it does use both cascading style sheets and dynamic HTML. It is an ongoing struggle to support the browsers as they evolve.

Professional astronomers generally have a good command of English, but SkyServer supports an international user community that includes children and non-scientists. So, the web page hierarchy branches three ways: there is an English branch, a German branch, and a Japanese branch. Other languages can be added by people fluent in those languages. Each mirrored site will have all the data and supports all the languages.

6. SkyServer for Education

The public access to real astronomical data and the SkyServer's web interfaces are a resource for science education and public outreach. Today, most students learn astronomy through textbook exercises that use artificial data or data that was taken centuries ago. With SkyServer, students can study data from galaxies never before seen by human eyes. We are designing several interactive educational projects that let students use SkyServer to learn astronomy and computational science concepts.

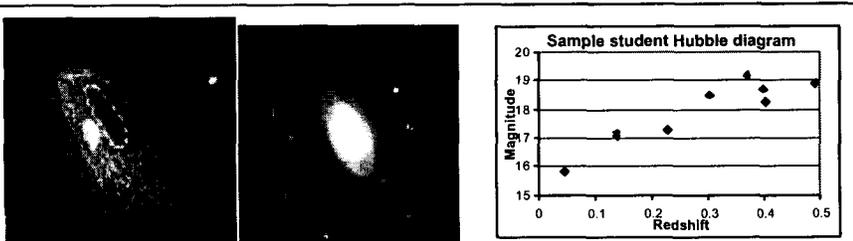


Figure 4: An example from the "Old Time Astronomy" project compares the sketch of Galaxy M64 made by amateur astronomer Michael Geldorp (left) to the image of the same galaxy from the Digitized Sky Survey (right). A more advanced project has students plot a Hubble diagram at right (showing redshift and relative distance of nine galaxies) to "discover" the expansion of the universe.

The educational projects address two audiences: first, bright students excited about astronomy who want to work with data independently, and second, students taking general astronomy or other science courses as part of a school curriculum. To accommodate both audiences, we offer several different project levels, from “For Kids” (projects for elementary school students) to “Challenges” (projects designed to stretch bright college undergraduates). All projects designed for use in schools include a password-protected teachers’ site with solutions, advice on how to lead classes through projects and correlations to national education standards [Project 2061].

For example, a kids’ project, “Old Time Astronomy,” (<http://skyserver.sdss.org/en/proj/kids/oldtime/>) asks students to imagine what astronomy was like before the camera was invented, when astronomers had to record data through sketches. The project shows SDSS images of stars and galaxies, and then asks students to sketch what they see. After a student has sketched the image, she trades with another student to see if the other student can guess which image was sketched (Figure 4.)

A project for advanced high school students and college undergraduates explores the expanding universe. The web site first gives students background reading about how scientists know the universe is expanding. Then, it lets students discover the expansion for themselves by making a *Hubble Diagram* – a plot of the velocities (or redshifts) of distant galaxies as a function of their distances from Earth. A sample student Hubble diagram is shown in Figure 4. Among other things, this teaches students how to work with real data.

About 100 hours of lessons are online now. Many more exercises and projects are being developed around the SkyServer. One particularly successful one was done by a teacher and some students in Mexico – there is growing international interest in using the SDSS to teach science to students in their native language (Spanish in that case).

One of the most exciting aspects of using SkyServer in education is its potential for students to pose and answer groundbreaking astronomical research questions. Because students can examine exactly the same data as professional astronomers, they can ask the same questions. Each school project ends with a “final challenge” that invites students to do independent follow-up work on a question that interests them. We are also working on a mentorship program that will match students working on school science fair projects with professional astronomers that volunteer to act as mentors, helping students to refine their research questions and to obtain the data they need to find answers.

7. Site Traffic

The SkyServer has been operating since June 2001. In the first 7 months it served about 2.5 million hits, a million page views via 70 thousand sessions. About 4% of these are to the Japanese sub-web and 3% to the German sub-web. The educational projects got about 8% of the traffic: about 250 page views a day. The server has been up 99.83% of the time. There have been 14 reboots, 8 to for software upgrades and 5 associated with failing power. The patches cause outages of 5 minutes, the power and operations outages last several hours. Not shown in the statistics, but clearly visible in Figure 5 are two network outages or overloads that plagued Fermilab on 22 June and 26 July. Conversely,

the peak traffic coincided with classes using the site, news articles mentioning it, or with demonstrations at Astronomy conferences. The sustained usage is about 500 people accessing about 4,000 pages per day. The site is configured to handle a load 100x larger than that. A TV show on October 2, generated a peak 20x the average load. About 30% of the traffic is from other sites “crawling” the SkyServer – extracting the data and images. There are about 5 “hacker attacks” per day.

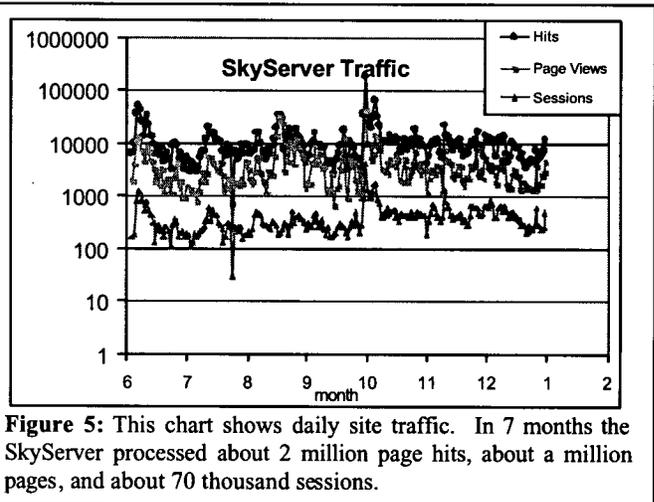


Figure 5: This chart shows daily site traffic. In 7 months the SkyServer processed about 2 million page hits, about a million pages, and about 70 thousand sessions.

8. Web Server Deployment & Administration

The application is primarily administered from Johns Hopkins and San Francisco using the Windows™ remote windows system (Terminal Server) feature. The Fermilab staff manages the physical hardware, the network, and site security. There is a mirror server at Johns Hopkins for incremental development and testing. The two sites are synchronized about once per week.

9. The Data and Databases

The SDSS processing pipeline at Fermilab examines the 5-color images from the telescope and identifies *photo objects* as either *stars*, *galaxies*, *trail* (cosmic ray, satellite,...), or some *defect*. The classification is probabilistic; it is sometimes difficult to distinguish a faint star from a faint distant small galaxy. In addition to the basic classification, the pipeline extracts about 400 attributes from an object, including a “cutout” of the object’s pixels in the 5 color bands.

The actual observations are taken in stripes about 2.5° wide and 120° long (see Figure 6). To further complicate things, these stripes are in fact the mosaic of two night’s observations (two strips) with about 10% overlap. The stripes themselves have some overlaps near the horizon. Consequently, about 11% of the objects appear more than once in the pipeline. The pipeline picks one object instance as *primary* but all instances are recorded in the database. Even more challenging, one star or galaxy often overlaps another, or a star is part of a cluster. In these cases *child* objects are *deblended* from the parent object, and each child also appears in the database (deblended parents are never primary.) In the end about 80% of the photo objects are primary.

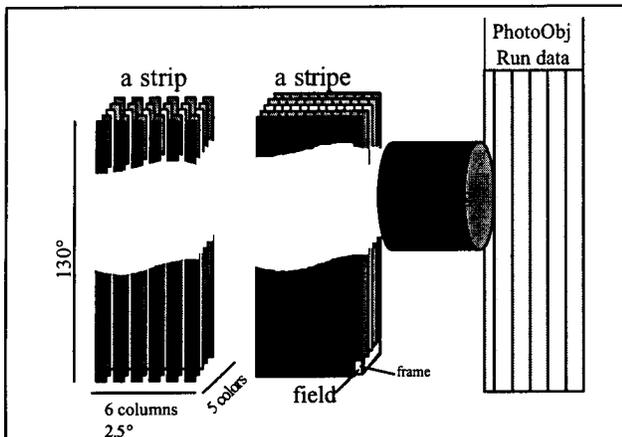


Figure 6: The survey merges two observations (two interleaved strips from two nights) into a stripe. Each strip (and hence each stripe) observes the sky in 5 different optical bands (colors). The stripe is processed by the pipeline to produce the images and photo objects.

The photo objects have positional attributes - right ascension and declination in the J2000 coordinate system, also represented as the Cartesian components of a unit vector, and an index into a Hierarchical Triangular Mesh (HTM). They also have brightness stored in logarithmic units (magnitudes) with error bars in each of the five color bands. These magnitudes are measured in six different ways (for a total of 60 attributes). The image processing pipeline also measures each galaxy's extent in several ways in each of the 5 color bands with error estimates. The pipeline assigns about a hundred additional properties to each object - these attributes are variously called flags, status, and type and are encoded as bit flags.

The pipeline tries to correlate each object with objects in other surveys: United States Naval Observatory [USNO], Röntgen Satellite [ROSAT], Faint Images of the Radio Sky at Twenty-centimeters [FIRST], and others. Successful correlations are recorded in a set of relationship tables.

Spectrograms are the other data product produced by the Sloan Digital Sky Survey. About 600 spectra are observed at once using a single plate with optical fibers going to different CCDs. The pipeline processing typically extracts about 30 spectral lines from each spectrogram and carefully estimates the object's redshift.

9.1. The Relational Database Design

Originally, the SDSS developed the entire database on ObjectivityDB™ [SDSS-Science Archive]. The designers used subclasses extensively: for example the PhotoObject has Star and Galaxy subclasses. ObjectivityDB supports arrays so the 5 colors naturally mapped to vectors of 5 values. Connections to parents, children, spectra, and to other surveys were represented as object references. Translating the ObjectivityDB™ design into a relational schema was not straightforward; but we wanted to preserve as much of the original design as possible in order to preserve the existing knowledge, skills, and documentation.

The SQL relational database language does not support pointers, arrays, or sub-classing - it is a much simpler data model. This is

both a strength and a liability. We approached the SQL database design by using views for subclassing, and by using foreign keys for relationships.

9.1.1. Photographic Objects

Starting with the imaging data, the PhotoObj table has all the star and galaxy attributes. The 5 color attribute arrays and error arrays are represented by their names (u, g, r, i, z.) For example, Model-Mag_r is the name of the "red" magnitude as measured by the best model fit to the data. In cases where names were unnatural (for example in the profile array) the data is encapsulated by access functions that extract the array elements from a blob. Pointers and relationships are represented by "foreign keys".

The result is a snow-flake schema with the photoObj table in the center other tables clustered about it (Figure 7). The 14 million photoObj records each have about 400 attributes describing the object - about 2KB per record. The Field table describes the processing that was used for all objects in that field, in all frames. The other tables connect the PhotoObj table to literals (e.g. flags & fPhotoFlags('primary')), or connect the object to objects in other surveys. One table, neighbors, is computed after the data is loaded. For every object the neighbors table contains a list of all other objects within 1/2 arcminute of the object (typically 10 objects). This speeds proximity searches.

9.1.2. Spectroscopic Objects

Spectrograms are the second kind of data object produced by the Sloan Digital Sky Survey. About 600 spectra are observed at once using a single plate with optical fibers going to two different spectrographs. The plate description is stored in the plate table, and the description of the spectrogram is stored in the specObj table (each spectrogram has a handsome GIF image associated with it.). The pipeline processing typically extracts about 30 spectral lines from each spectrogram. The spectral lines are stored in the SpecLine table. The SpecLineIndex table has quantities derived from analyzing line groups. These quantities are used by astronomers to characterize the types and ages of astronomical objects. Each line is cross-correlated with a model and corrected for redshift. The resulting attributes are stored in the xCRedShift table. A separate redshift is derived using only emission lines. Those quantities are stored in the eLRedShift table. All these tables are integrated with foreign keys - each specObj object has a unique specObjID key, and that same key value is stored as part of the key of every related spectral line. To find all the spectral lines of object 512 one writes the query

```
select *
from specLine
where specObjID = 512
```

The spectrographic tables also form a snowflake schema that gives names for the various flags and line types. Foreign keys connect PhotoObj and SpecObj tables if a photo object has a measured spectrogram.

9.1.3. Database Access Design - Views, Indices, and Access Functions

The PhotoObj table contains many types of objects (primaries, secondaries, stars, galaxies,...). In some cases, users want to see

all the objects; but typically users are just interested in primary objects (best instance of a deblended child), or they want to focus on just Stars, or just Galaxies. So, views are defined on the PhotoObj table (views are virtual tables defined by queries on the base table):

```
photoPrimary: PhotoObj with flags('primary' & 'OK run')
Star: photoPrimary with type='star'
Galaxy: photoPrimary with type='galaxy'
```

Most users work in terms of these views rather than the base table. This is the equivalent of sub-classing. The SQL query optimizer rewrites such queries so that they map down to the base *photoObj* table with the additional qualifiers.

To speed access, the PhotoObj table is heavily indexed (these indices also benefit the views). In a previous design based on an object-oriented database ObjectivityDB™ [Thakar], the architects replicated vertical data slices, called *tag* tables, which contain the most frequently accessed object attributes. These tag tables are about ten times smaller than the base tables (a few hundred1 bytes rather than a few thousand bytes).

Our concern with the tag table design is that users must know which attributes are in a tag table and must know if their query is *covered* by the fields in the tag table. Indices are an attractive alternative to tag tables. An index on fields A, B, and C gives an automatically managed tag table on those 3 attributes plus the primary key – and the SQL query optimizer automatically uses that index if the query is covered by (contains) those fields. So, indices perform the role of tag tables and lower the intellectual load on the user. In addition to giving a column subset that speeds sequential scans by ten to one hundred fold, indices also cluster data so that range searches are limited to just one part of the object space. The clustering can be by type (star, galaxy), or space, or magnitude, or any other attribute. One limitation is that Microsoft's SQL Server 2000 limits indices to 16 columns.

Today, the SkyServer database has tens of indices, and more will be added as needed. The nice thing about indices is that they speed up any queries that can use them. The downside is that they slow down the data insert process – but so far that has not been a problem. About 30% of the SkyServer storage space is devoted to indices.

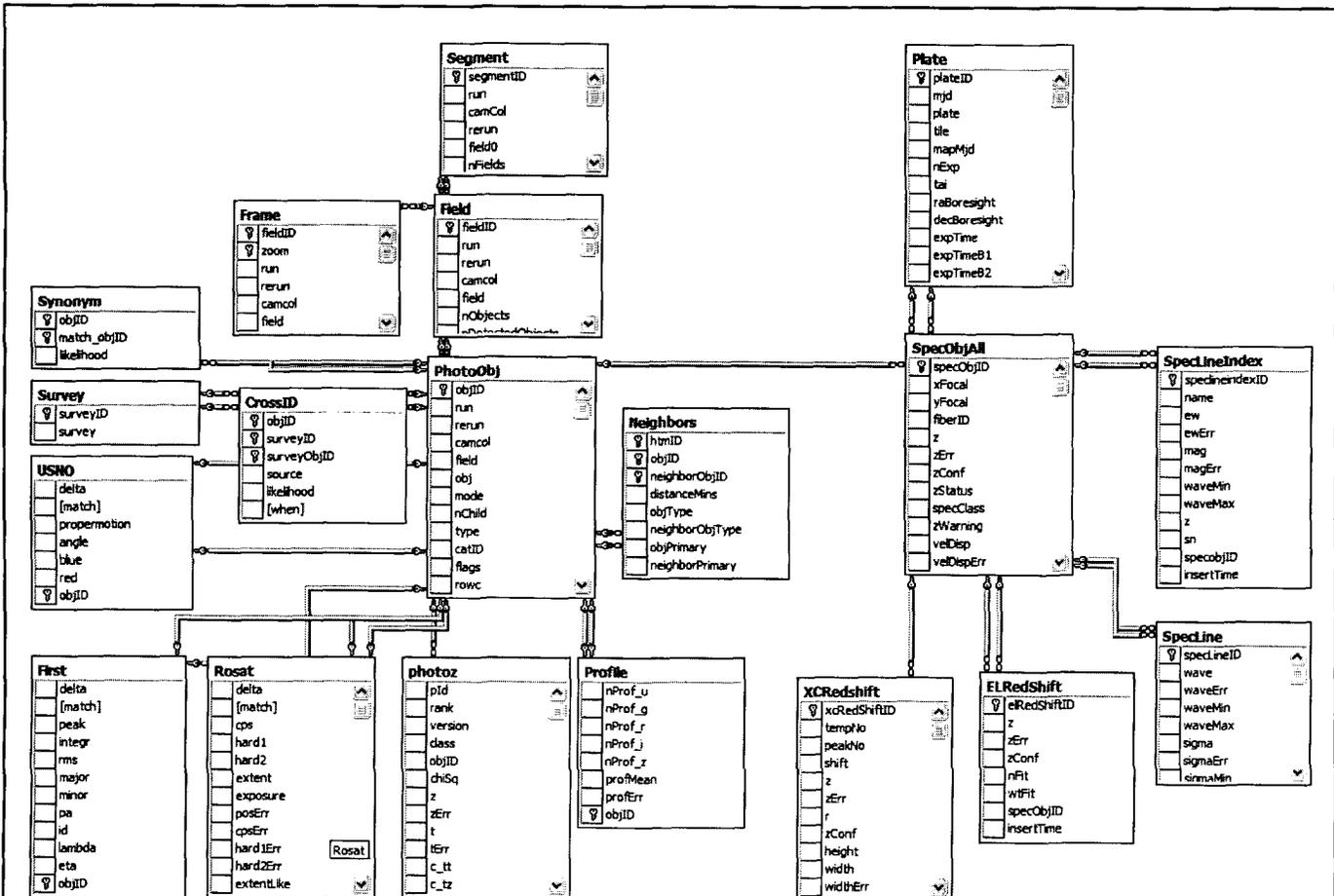


Figure 7: The schema for photographic objects like stars and galaxies is shown at left. Observations are processed in *fields*. Each field in turn contains many *objects*. Objects have an *image* and a *profile* array, giving the brightness in concentric rings around the object. The spectrographic snowflake schema is shown at right. Each *plate* produces about 800 *spectra* that in turn each have about 30 spectral *lines*. Lines are further analyzed (*line-index*) and corrected for redshift. Correlations to other surveys (Rosat, First, USNO, ...) are recorded in the tables at left. The schema is documented online at <http://skyserver.sdss.org/en/help/docs/browser.asp>.

In addition to the indices, the database design includes a fairly complete set of foreign key declarations to insure that every profile has an object; every object is within a valid field, and so on. We also insist that all fields are non-null. These integrity constraints are invaluable tools in detecting errors during loading and they aid tools that automatically navigate the database.

Table 1: Count of records and bytes in major tables. Indices approximately double the space.

Table	Records	Bytes
Field	14k	60MB
Frame	73k	6GB
PhotoObj	14m	31GB
Profile	14m	9GB
Neighbors	111m	5GB
Plate	98	80KB
SpecObj	63k	1GB
SpecLine	1.7m	225MB
SpecLineIndex	1.8m	142MB
xcRedShift	1.9m	157MB
elRedShift	51k	3MB

9.1.4. Spatial Data Access

The SDSS scientists are especially interested in the galactic clustering and large-scale structure of the universe. In addition, the web interface routinely asks for all objects in a certain rectangular or circular area of the celestial sphere.

The SkyServer uses three different coordinate systems. First right-ascension and declination (comparable to latitude-longitude in terrestrial coordinates) are ubiquitous in astronomy. The (x,y,z) unit vector in J2000 coordinates is stored to make arc-angle computations fast. The dot product and the Cartesian difference of two vectors are quick ways to determine the arc-angle or distance between them.

To make spatial area queries run quickly, the Johns Hopkins *hierarchical triangular mesh* (HTM) code [HTM, Kunszt1] was added to SQL Server. Briefly, HTM inscribes the celestial sphere within an octahedron and projects each celestial point onto the surface of the octahedron. This projection is approximately iso-area.

HTM partitions the sphere into the 8 faces of an octahedron. It then hierarchically decomposes each face with a recursive sequence of triangles –each level of the recursion divides each triangle into 4 sub-triangles (Figure 8). In SDSS’s 20-deep HTMs individual triangles are less than 0.1 arcseconds on a side. The HTM ID for a point very near the north pole (in galactic coordinates) would be something like 3,0,....,0. There are basic routines to convert between (ra, dec) and HTM coordinates.

These HTM IDs are encoded as 64-bit integers. Importantly, all the HTM IDs within the triangle 6,1,2,2 have HTM IDs that are between 6,1,2,2 and 6,1,2,3. So,

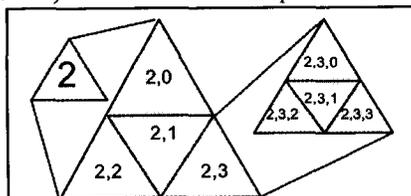


Figure 8: A Hierarchical Triangular Mesh (HTM) recursively assigns a number to each point on the sphere. Most spatial queries use the HTM index to limit searches to a small set of triangles. An HTM index is built as an extension of SQL Server’s B-trees.

a B-tree index of HTM IDs provides a quick index for all the objects within a given triangle. The HTM library is an SQL extended stored procedure wrapped in a table-valued function `spHTM_Cover(<area>)`. The `<area>` can be either a circle (`ra, dec, radius`), a half-space (the intersection of planes), or a polygon defined by a sequence of points. The function returns a table containing a row with start and end of an HTM triangle. The union of these triangles covers the specified area. One can join this table with the `PhotoObj` table to get a spatial subset of photo objects.

The `spHTM_Cover()` function is too primitive for most users, they actually want the objects nearby a certain object, or they want all the objects in a certain area. So, simpler functions are also supported. For example: `fGetNearestObjEq(1,1,1)` returns the nearest object within one arcminute of equatorial coordinate $(1^\circ, 1^\circ)$. These procedures are frequently used in queries and in the website access pages.

9.1.5. Summary of Database Design

In summary, the logical database design consists of photographic and spectrographic objects. They are organized into a pair of snowflake schemas. Subsetting views and many indices give convenient access to the conventional subsets (stars, galaxies, ...). Procedures and indices are defined to make spatial lookups convenient and fast.

9.2. Physical Database Design

The SkyServer initially took a simple approach to database design – and since that worked, we stopped there. The design counts on the SQL storage engine and query optimizer to make all the intelligent decisions about data layout and data access.

The data tables are all created in one file group. The database files are spread across 4 mirrored volumes. Each of the 4 volumes holds a database file that starts at 20 GB and automatically grows as needed. The log files and temporary database are also spread across these disks. SQL Server stripes the tables across all these files and hence across all these disks. It detects the sequential access, creates the parallel prefetch threads, and uses multiple processors to analyze the data as quickly as the disks can produce it. When reading or writing, this automatically gives the sum of the disk bandwidths (up to 140 MBps) without any special user programming.

Beyond this file-group striping, SkyServer uses all the SQL Server default values. There is no special tuning. This is the hallmark of SQL Server – the system aims to have “no knobs” so that the out-of-the box performance is quite good. The SkyServer is a testimonial to that goal. A later section discusses the hardware and the system performance.

9.4. Database Load Process

The SkyServer is a data warehouse: new data is added in batches, but mostly the data is queried. Of course these queries create intermediate results and may deposit their answers in temporary tables, but the vast bulk of the data is read-only.

Occasionally, a new schema is loaded (we are on V3 right now), so the disks were chosen to be large enough to hold three complete copies of the database.

From the SkyServer administrator's perspective, the main task is data loading -- which includes data validation. When new photo objects or spectrograms come out of the pipeline, they have to be added to the database. We are the system administrators -- so we wanted this loading process to be as automatic as possible.

The SDSS data pipeline produces FITS files, but also produces comma-separated list (csv) files of the object data and PNG files. The PNG files are converted to JPEG at various zoom levels, and an image pyramid is built before loading. These files are then copied to the SkyServer. From there, a script loads the data using the SQL Server's Data Transformation Service. DTS does both data conversion and the integrity checks. It also recognizes file names in some fields, and uses those names to place the contents of the corresponding image file (JPEG) as a blob field of the record. There is a DTS script for each table load step. In addition to loading the data, these DTS scripts write records in a *loadEvents* table recording the load time, the number of records in the source file, and the number of inserted records. The DTS steps also write trace files indicating the success or errors in the load step. A particular load step may fail because the data violates foreign key constraints, or because the data is invalid (violates integrity constraints.) In the web interface helps the operator to (1) undo the load step, (2) diagnose and fix the data problem, and (3) re-execute the load on the corrected data.

Loading runs at about 5 GB per hour (data conversion is very cpu intensive), so the current SkyServer data loads in about 12 hours.

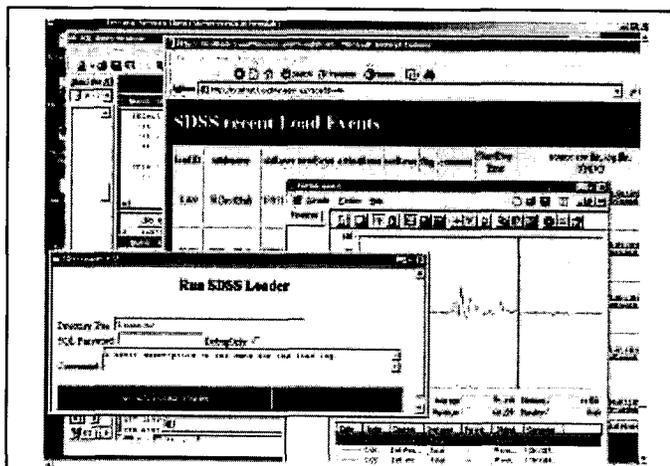


Figure 9: A screen shot of the SkyServer Database operations interface. The SkyServer is operated via the Internet using Windows2000 Terminal Server, a remote desktop facility built into the operating system. Both loading and software maintenance are done in this way. This screen shot shows a window into the backend system after a load step has completed. It shows the loader utility, the load monitor, a performance monitor window and a database query window. This remote operation has proved a godsend, allowing the Johns Hopkins, Microsoft, and Fermilab participants to manage the system from their offices.

A simple web user interface displays the load-events table and makes it easy to examine the CSV file and the load trace file. If the input file is easily repaired, that is done by the administrator, but often the data needs to be regenerated. In either case the first step is to UNDO the failed load step. Hence, the web interface has an UNDO button for each step.

The UNDO function works as follows: Each table in the database has a timestamp field that tells when the record was inserted (the field has *Current_Timestamp* as its default value.) The load event record tells the table name and the start and stop time of the load step. Undo consists of deleting all records of that table with an insert time between the bad load step start and stop times.

10. Personal SkyServer

A 1% subset of the SkyServer database (about .5 GB SQL Server database) can fit on a CD or be downloaded over the web. This includes the web site and all the photo and spectrographic objects in a 6° square of the sky. This personal SkyServer fits on laptops and desktops. It is useful for experimenting with queries, for developing the web site, and for giving demos. Essentially, any classroom can have a mini-SkyServer per student. With disk technology improvements, a large slice of the public data will fit on a single disk by 2003.

11. Data Mining the SkyServer Database

As explained in Section 3, the SkyServer database was designed to quickly answer the 20 queries posed in [Szalay]. The web server and service, and the outreach efforts came later. We were very pleased to find the 20 queries all have fairly simple SQL equivalents. Often the query can be expressed as a single SQL statement. In some cases, the query is iterative, the results of one query feeds into the next. These queries correspond to typical tasks astronomers would do with a TCL script driving a C++ program, extracting data from the archive, and then analyzing it. Traditionally most of these queries would have taken a few days to write in C++ and then a few hours or days to run against the binary files. So, being able to do the query simply and quickly is a real productivity gain for the Astronomy community.

This section examines some queries in detail. The first query (Query 1 in [Szalay]) is to find all galaxies without saturated pixels within 1' of a given point. Translated, sometimes the CCD camera is looking at a bright object and the cells are saturated with photons. Such data is suspect, so the queries try to avoid objects that have saturated pixels. So, the query uses the *Galaxy* view to subset the objects to just galaxies. In addition, it only considers pixels: `flags&fPhotoFlags('saturated')=0` objects. The last restriction is that the galaxy be nearby a certain spot. Astronomers use the J2000 right ascension and declination coordinate system. As explained in section 9.1.4, the SkyServer has some spatial data access functions that return a table of HTM ranges that cover an area. A second layer of functions return a table containing all the objects within a certain radius of a point. `fGetNearbyObjEq(185, -0.5, 1)` returns the IDs of all objects within 1 arcminute of the (ra,dec) point (185, -.5). The full query is then:

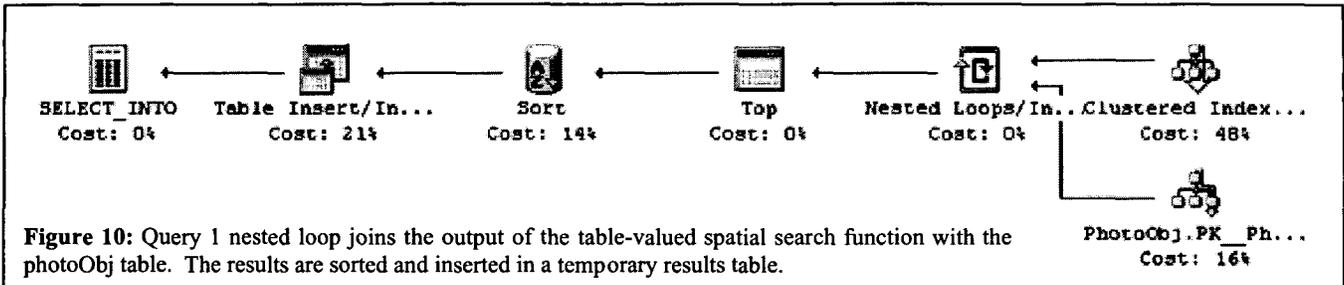


Figure 10: Query 1 nested loop joins the output of the table-valued spatial search function with the photoObj table. The results are sorted and inserted in a temporary results table.

```

declare @saturated bigint;
set @saturated = dbo.fPhotoFlags('saturated');
select G.objID, GN.distance
into ##results
from Galaxy as G
join fGetNearbyObjEq(185,-0.5, 1) as GN
on G.objID = GN.objID
where (G.flags & @saturated) = 0
order by distance

```

This query returns 19 galaxies in 50 milliseconds of CPU time and 0.19 seconds of elapsed time. Figure 10 shows the query plan (the table-valued function `GetNearbyObjEq()` is nested-loop joined with the `photoObj` table – each row from the function is used to probe the `photoObj` table to test the saturated flag, the primary object flag, and the galaxy type.). The function returns 22 rows that are joined with the `photoObj` table on the `ObjID` primary key to get the object’s flags. 19 of the objects are not saturated and are primary galaxies, so they are sorted by distance and inserted in the results temporary table.

Another typical access pattern is to do a complex “color cut” of the object space: looking for objects with certain properties characteristic of a quasar or active galactic nuclei. Interestingly, these queries look very complex, but they are really a table scan with a very complex predicate. As such they are limited by the disk speed (the processors can evaluate millions of predicates per second). We refer the reader to the archival paper [Gray] for examples of this (see queries 5, 14 19, 20.)

To give a more interesting example, Query 15 was “find all asteroids.” Objects are classified as asteroids if their positions change over the time of observation. SDDS makes 5 successive observations from the 5 color bands over a 5 minute period. If an object is moving, the successive images see a moving image against the fixed background of the galaxies. The processing pipeline computes this movement velocity as `rowV` (the row velocity) and `colV` the column velocity. Extremely high or negative velocities represent errors, and so should be ignored.

The query is a table scan computing the velocities and selecting objects that have high (but reasonable) velocity.

```

select objID,
sqrt(rowv*rowv+colv*colv) as velocity,
dbo.fGetUrlExpId(objID) as Url
into ##results
from PhotoObj
where (rowv*rowv+colv*colv) between 50 and 1000
and rowv >= 0 and colv >= 0

```

SQL Server selects a parallel sequential scan of the `PhotoObj` table (there is no covering index.) The query uses 72 seconds of CPU time in 162 second of elapsed time to evaluate the predicate on each of the 14M objects. It finds 1,303 candidates. Figure 11 shows the query plan and also a picture of one of the 4 objects discovered by the query (the query returns the URL and objectID of each object so that it can be easily examined).

These are “slow moving” objects. A different query is needed to find fast moving objects (based on a tcl script written by Steve Kent). This query looks for streaks in the sky that line up. These streaks are not close enough to be identified by the image processing pipeline as a single object. The query examines all pairs of objects in a given area (`run, camcol, field`) that are elongated and have a magnitude between 6 and 22. It selects the red and green candidates, by requiring that they are fainter in all the other colors. Next the query joins on these two sets requiring that the pair’s magnitudes in `g` and `r` are within 2, and that the images are within 4 arcminutes of one another (in the same `run` and `camcol`, but perhaps in adjacent fields.) The query found four such pairs, in one of them the red object is degenerate, probably deblended. Each of the other three is a NEO (near earth object).

The sql query optimizer chooses an index scan (since there is a covering index for the attributes. It does a nested loops join of the red and green candidate objects seeing if their ellipticities “line up.” For each object that qualifies in the red band, finding all the qualifying green objects. When the query finds a matching pair, it checks to see if they have comparable magnitudes (differ by at most 2)). Using the index, the query finds 4 objects in 55 seconds elapsed and 51 seconds of CPU time. Without the index the query takes about 10 minutes—since it is nested-loops join of two table scans.

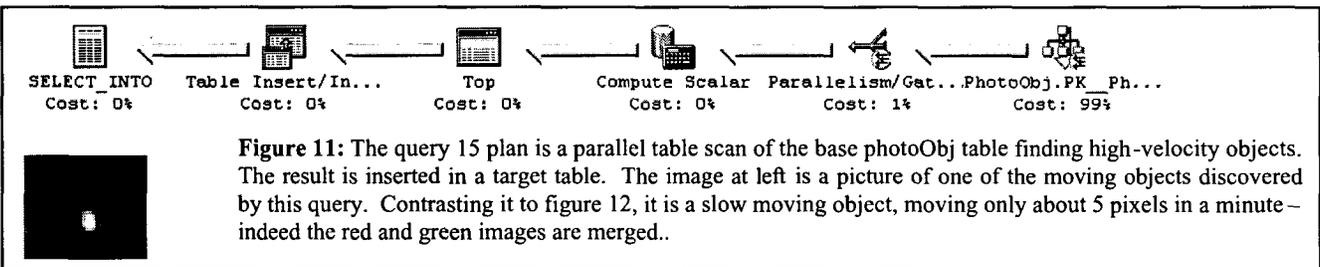


Figure 11: The query 15 plan is a parallel table scan of the base `photoObj` table finding high-velocity objects. The result is inserted in a target table. The image at left is a picture of one of the moving objects discovered by this query. Contrasting it to figure 12, it is a slow moving object, moving only about 5 pixels in a minute – indeed the red and green images are merged..

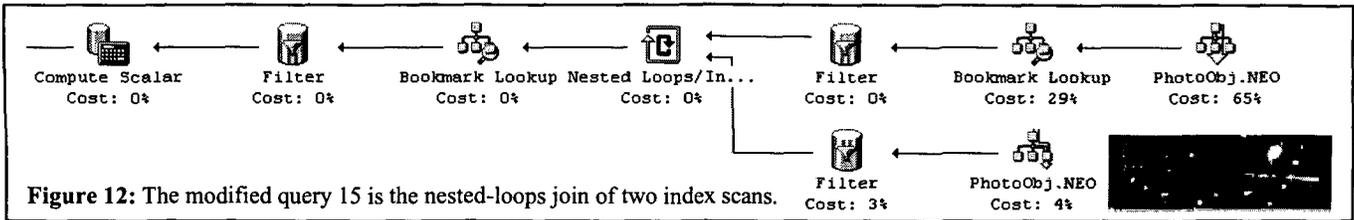


Figure 12: The modified query 15 is the nested-loops join of two index scans.

```
Select r.objID as rId, g.objID as gId,
      dbo.fGetUrlExpId (r.objID) as rURL,
      dbo.fGetUrlExpId (g.objID) as gURL
from PhotoObj r, PhotoObj g
where r.run = g.run and r.camcol=g.camcol
      and abs(g.field-r.field) <= 1
      and ((power(r.q_r,2) + power(r.u_r,2)) >
           0.111111) -- q/u is ellipticity
-- the red selection criteria
      and r.fiberMag_r between 6 and 22
      and r.fiberMag_r < r.fiberMag_u
      and r.fiberMag_r < r.fiberMag_g
      and r.fiberMag_r < r.fiberMag_i
      and r.fiberMag_r < r.fiberMag_z
      and r.parentID=0
      and r.isoA_r/r.isoB_r > 1.5
      and r.isoA_r > 2.0
-- the green selection criteria
      and ((power(g.q_g,2) + power(g.u_g,2)) >
           0.111111) -- q/u is ellipticity
      and g.fiberMag_g between 6 and 22
      and g.fiberMag_g < g.fiberMag_u
      and g.fiberMag_g < g.fiberMag_r
      and g.fiberMag_g < g.fiberMag_i
      and g.fiberMag_g < g.fiberMag_z
      and g.parentID=0
      and g.isoA_g/g.isoB_g > 1.5
      and g.isoA_g > 2.0
-- the match-up of the pair
--(note acos(x) ~ x for x-1)
      and sqrt(power(r.cx-g.cx,2)
              +power(r.cy-g.cy,2) +power(r.cz-g.cz,2)) *
              (180*60/pi())) < 4.0
      and abs(r.fiberMag_r-g.fiberMag_g) < 2.0
```

These three examples give a flavor of how the SkyServer is used for data mining. Much more detail is found in [Gray].

Summarizing the other queries, many run in a few seconds. Some that involve a sequential scan of the database take about 3 minutes. One involves a spatial join and takes ten minutes. As the data grows from 60GB to 1TB, the queries will slow down by a factor of 20. Processors and disks will be at least three times faster then, but queries will be seven times slower on these larger datasets. So, future SkyServers will need more than 2 processors and more than 4 disks. By using CPU and disk parallelism,

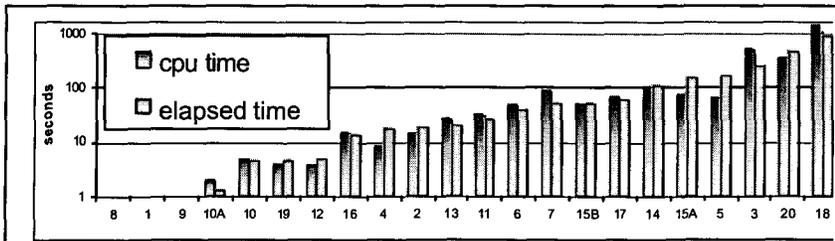


Figure 13: The query execution times (system in Figure 14). The system is disk limited in all cases. So 2x more disks would cut the time nearly in half. The detailed statistics are in [Gray].

it should be possible to keep response times in the “few minutes” range.

It takes both a good understanding of astronomy, a good understanding of SQL, and a good understanding of the database to translate the queries into SQL. In watching how “normal” astronomers access the SX web site, it is clear that they use very simple SQL queries. It appears that they use SQL to extract a subset of the data and then analyze that data on their own system using their own tools. SQL, especially complex SQL involving joins and spatial queries, is just not part of the current astronomy toolkit.

Indeed, our actual query set includes 15 additional queries posed by astronomers using the Objectivity™ archive at (<http://archive.stsci.edu/sdss/software/>). Those 15 queries are much simpler and run more quickly than most of the original 20 queries.

A good visual query tool that makes it easier to compose SQL would ameliorate part of this problem, but this stands as a barrier to wider use of the SkyServer by the astronomy community. Once the data is produced, there is still a need to understand it. We have not made any progress on the problem of data visualization.

It is interesting to close with an anecdote about the use of the SkyServer for data mining. Most objects do not have spectrograms. Only 1% are targeted for spectroscopy. Hence, 99% of the objects do not have measured redshifts – and redshifts are a surrogate for distance. As it turns out, an objects’ redshift can be estimated by its 5-color optical measurements. These estimates are surprisingly good [Budavari1, Budavari2]. However, the estimator requires a training set. There was a part of parameter space – where only 3 galaxies were in the training data and so the estimator did a poor job. To improve the estimator, we wanted to measure the spectra of 1,000 such galaxies. Doing that required designing some plates that measure the spectrograms. The plate drilling program is huge and not designed for this task. We were afraid to touch it. But, by writing some SQL and playing with the data, we were able to develop a drilling plan in an evening. Over the ensuing 2 months the plates were drilled, used for observation, and the data was reduced. Within an hour of getting the data, they were loaded into the SkyServer database and we have used them to im-

prove the redshift predictor — it became much more accurate on that class of galaxies. Now others are asking our help to design specialized plates for their projects.

We believe this and many similar experiences are a very promising sign that commercial database tools can indeed help scientists organize their data for data mining and easy access.

12. Hardware Design and Performance

The SkyServer can run on a single processor system with just one large disk, but the SkyServer at Fermilab runs on more capable hardware donated by Compaq Computer (Figure 14.)

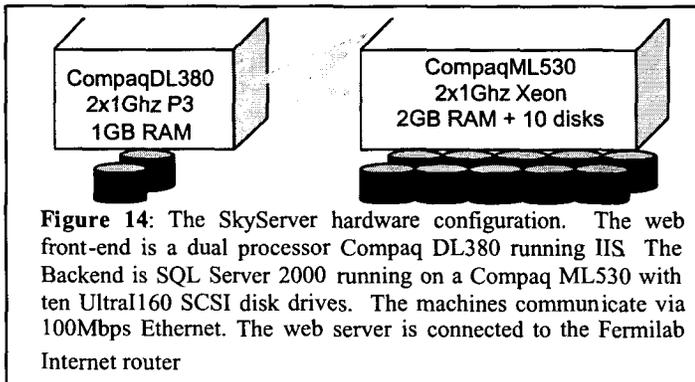
The web server runs Windows2000 on a Compaq™ DL380 with dual 1GHz PentiumIII processors. It has 1GB of 133MHz SDRAM, a 64-Bit/66MHz Single Channel Ultra3 SCSI Adapter with a mirrored disk. This web server does almost no disk IO during normal operation, but we clocked the disk subsystem at over 30MBps. The web server is also a firewall, it does not do routing and it has a separate “private” 100Mbps Ethernet link to the backend database server.

Most data mining queries are IO-bound, so the database server is configured to give fast sequential disk bandwidth, healthy CPU power, and high availability. The database server is a Compaq ProLiant ML530 running SQL Server 2000 and Windows2000. It has two 1GHz Pentium III Xeon processors, 2GB of 133MHz SDRAM; a 2-slot 64bit/66MHz PCI bus, a 5-slot 64bit/33MHz PCI, and a 32bit PCI bus. It has two 64-Bit/66MHz Single Channel Ultra3 SCSI Adapters. The ML530 also has a complement of high-availability hardware components: redundant, hot-swappable power supplies, fans, and SCSI disks.

The server has ten Compaq 37GB 10K rpm Ultra160 SCSI disks, five on each SCSI channel. Windows2000's software RAID manages these as five mirrors (RAID1's): one for the operating system and software, and the remaining four for databases. The database filegroups (both data and log) are spread across these four mirrors. SQL Server stripes the data across the four volumes, effectively managing the data disks as a RAID10 (striping plus mirroring). This configuration can scan data at 140 MBps for a simple query like:

```
select count(*)
from photoObj
where (r-g)>1.
```

[Gray] gives detailed timings on twenty complex queries, but to summarize: a typical index lookup runs primarily in memory and completes within a second or two. SQL Server uses available memory to cache frequently used data. Index scans of the 14M row photo table run in 7 seconds “warm” (5 m records per second when cpu bound), and 17 seconds cold, on a 4-disk 2-cpu Server. Queries that scan the entire 30GB photoObj table run at about 140 MBps and so take about 3 minutes. These scans use the available CPUs and disks in parallel – but are IO bound.



When the SkyServer project began, the existing software was delivering 0.5 MBps and heavy CPU consumption – indeed that was the main reason for considering a relational database over ObjectivityDB™. The SkyServer goal was 50MBps at the user level on a single machine. Beyond that, arrays of parallel database machines could deliver more bandwidth. This would give 5-second response to simple queries, and 5-minute response to full database scans. As it stands SQL Server and the Compaq hardware exceeded the performance goals by 500%. In general 4-disk workstation-class machines run at the 70MBps PCI (32/33) bus limit, while 8-disk RAID0 server-class machines run at 300 MBps. As the SDSS data grows, arrays of more powerful machines should allow the SkyServer to return most answers within seconds or minutes depending on whether it is an index search, or a full-database scan.

Various disk-controller configurations were measured to test maximum IO speed (see Figure 15):

- The memory speed is about 600 MBps (single-threaded read: 590 MB/s, write: 590 MB/s, and copy: 590 MB/s; multi-threaded read: 849MB/s, write: 374 MB/s, and copy: 300 MB/s.)
- A disk delivers about 40 MBps (between 37 and 51 Mbps). Three disks on one ultra3 controller deliver up to 119 MBps (99.8% scale up.)
- Three disks saturate an ultra3 controller; beyond that additional controllers are needed.
- A 64-bit/33MHz PCI bus saturates at about 220MBps – not quite enough for two ultra3 controllers.
- A 12-disk 4-controller software RAID0 system can scan NTFS files at 430 megabytes per second with 12% cpu utilization.
- SQL Server runs a minimal (`select count(*)`) query at 331MBps on nine disks on three controllers.
- At that rate, SQL is evaluating 2.6 million 128-byte tag records per second (mrps); and, the cpu utilization is 75% -- 10 clocks per byte (cpb), 1300 clocks per record (cpr).
- The more complex query `count(*) where (r-g)>1` is cpu bound, using 19 cpb and 2300 cpr.
- When the data is all in main memory, SQL scans at 5 mrps.

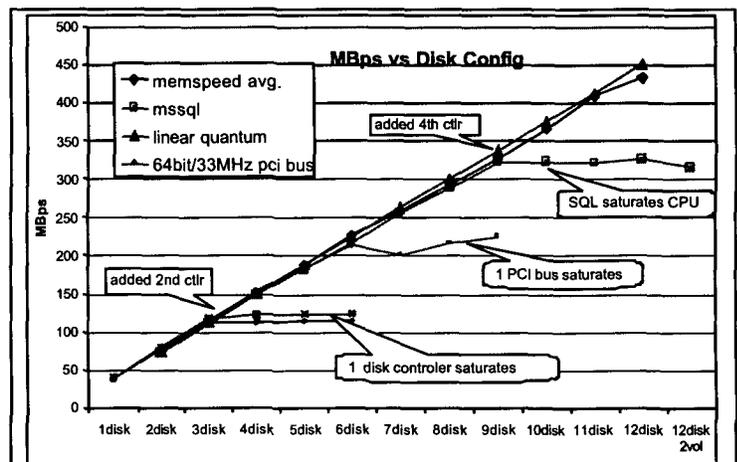


Figure 15: Sequential IO speed is important for data mining queries. This graph shows the sequential scan speed (megabytes per second) as more disks and controllers are added (one controller added for each 3 disks). It indicates that the SQL IO system process about 320MBps (and 2.7 million records per second) before it saturates.

13. Summary and Next Steps

The SkyServer is a web site and SQL database containing the Sloan Digital Sky Survey early data release (about 14 million celestial objects and 50 thousand spectra.) It is accessed over the Internet via a website that provides point-and-click access as well as several query interfaces including form-based reports, raw SQL queries, and a GUI query builder.

We designed the SkyServer both as a website for easy public access and as a data mining site. To test the data mining capabilities; we implemented 20 complex astronomy queries and evaluated their performance. We are pleased with the performance. Others can clone the server for a few thousand dollars. The data and web server are also being replicated at various astronomy and computer science institutes to either explore the data or to experiment with better ways to analyze and visualize the data.

The tools described here are a modest step towards the goal of providing excellent data analysis and visualization tools. We hope that the SkyServer will enable computer scientists to advance this field by building better tools. The SkyServer now has four main evolutionary branches:

Public SkyServer: New versions of the SDSS data will be released yearly. In addition, the SkyServer will get better documentation and tools as we get more experience with how it is used. There are Japanese and German clones of the website, and there are plans to clone it at several other sites. A basic server costs less than ten thousand dollars.

Virtual Observatory (VO): The SDSS is just one of many astronomy archives and resources on the Internet. The Internet will soon be the world's best telescope. It will have much of the world's astronomy data online covering all the measured spectral bands. The data will be cross-correlated with the literature. It will be accessible to everyone everywhere. And, if the VO is successful, there will be great tools to analyze all this data. The SkyServer is being federated with VirtualSky and NED at CalTech [VirtualSky] as Web Services, and with VizieR and Simbad at Strasbourg [VizieR, Simbad]. These are just the first steps towards a broader federation.

The Science Archive: A second group of SkyServers with preliminary (not yet released) data will form a virtual private network accessible to the SDSS consortium. These servers will have more sophisticated users who are intimately familiar with the data. These users have intimate knowledge of the data, and they perform careful data scrubbing and validation before its public release. So these servers will have unique needs.

Outreach and Curriculum Development: The SDSS data is a great vehicle for teaching both astronomy and computational science. The data is real -- everything comes with error bars, everything has a strong science component. The SDSS data also has strong graphical, spatial, and temporal components. It is fairly well documented and is public. And of course, it's big by today's standards. We hope that educators will "discover" the SkyServer and its educational potential -- both at K-12 and at the university level.

14. Acknowledgements

We acknowledge our obvious debt to the people who built the SDSS telescope, those who operate it, those who built the SDSS processing pipelines, and those who operate the pipeline at Fermilab. The SkyServer data depends on the efforts of all those people. Compaq and Microsoft generously donated hardware, software, and money to support the SkyServer. Tom Barclay advised us on the web site design, construction, and operation. Roy Gal, Steve Kent, Rich Kron, Robert Lupton, Steve Landy, Robert Sparks, Mark Subba Rao, Don Slutz, and Tamas Szalay contributed to the site's content and development. Sadanori Okamura, Naoki Yasuda, and Matthias Bartelmann built the Japanese and German versions of the site. Rosa Gonzalez and Kausar Yasmin helped with testing and developed additional class material.

15. References

- [Barclay] T. Barclay, D.R. Slutz, J. Gray, "TerraServer: A Spatial Data Warehouse," Proc. ACM SIGMOD 2000, pp: 307-318, June 2000
- [FIRST] Faint Images of the Radio Sky at Twenty-centimeters (FIRST) <http://sundog.stsci.edu>
- [FITS] FITS - Flexible Image Transport System, <http://heasarc.gsfc.nasa.gov/docs/heasarc/fits.html>
- [Gray] J. Gray, D. Slutz, A. Szalay, A. Thakar, J. vandenBerg, P. Kunszt, C. Stoughton, "Data Mining the SDSS SkyServer Database," Microsoft Technical Report MSR-T R-2002-01, Jan. 2002.
- [HTM] Hierarchical Triangular Mesh <http://www.sdss.jhu.edu/htm/>
- [Kunszt] P. Z. Kunszt, A. S. Szalay, I. Csabai, A. R. Thakar "The Indexing of the SDSS Science Archive" in *Proc ADASS IX*, eds. N. Manset, C. Veillet, D. Crabtree. (ASP Conference series), 216, 141-145 (2000)
- [Malik] SkyServer Query Analyzer, <http://skyserver.sdss.org/en/download/>
- [MAST] Multi Mission Archive at Space Telescope. <http://archive.stsci.edu/index.html>
- [Memspeed] A program to measure IO bandwidths on Windows: http://research.microsoft.com/BARC/Sequential_IO/memspeed.zip
- [NED] NASA/IPAC Extragalactic Database, <http://nedwww.ipac.caltech.edu/>
- [Project 2061] Principles and Standards, <http://www.project2061.org/>
- [ROSAT] Röntgen Satellite (ROSAT) <http://heasarc.gsfc.nasa.gov/docs/rosat/rass.html>
- [SDSS] D.G. York et al., The Sloan Digital Sky Survey: Technical Summary, The Astronomical Journal. 120 (2000) 1579-1587,
- [SDSS-EDR] C. Stoughton et. al., "Sloan Digital Sky Survey Early Data Release", The Astronomical Journal, 123 1:485-548 (2002)
- [SDSS-Science Archive] <http://www.sdss.jhu.edu/ScienceArchive/doc.html>
- [Simbad] SIMBAD Astronomical Database, <http://simbad.u-strasbg.fr/>
- [Szalay] A. Szalay, P. Z. Kunszt, A. Thakar, J. Gray, D. R. Slutz. "Designing and Mining Multi-Terabyte Astronomy Archives: The Sloan Digital Sky Survey," Proc. ACM SIGMOD 2000, pp451-462, 2000
- [Thakar] A.R. Thakar, P.Z. Kunszt, A.S. Szalay and G.P. Szokoly: "Multi-threaded Query Agent and Engine for a Very Large Astronomical Database," in *Proc ADASS IX*, eds. N. Manset, C. Veillet, D. Crabtree, (ASP Conference series), 216, 231 (2000).
- [USNO] US Naval Observatory <http://www.usno.navy.mil/products.shtml>
- [Virtual Sky] Virtual Sky, <http://VirtualSky.org/>
- [VizieR] VizieR Service, <http://vizier.u-strasbg.fr/viz-bin/VizieR>