Online Science
The World-Wide Telescope as a Prototype For the New Computational Science

Jim Gray
Microsoft Research
http://research.microsoft.com/~gray
Outline

• The World Wide Telescope Idea
• Data Mining the Sloan Digital Sky Survey
• Spherical Geometry in SQL
Computational Science

• Traditional Empirical Science
  – Scientist gathers data by direct observation
  – Scientist analyzes data

• Computational Science
  – Data captured by instruments
    Or data generated by simulator
  – Processed by software
  – Placed in a database / files
  – Scientist analyzes database / files
World Wide Telescope
Virtual Observatory

http://www.astro.caltech.edu/nvoconf/
http://www.voforum.org/

• Premise: Most data is (or could be) online.
• So, the Internet is the world’s best telescope:
  – It has data on every part of the sky
  – In every measured spectral band: optical, x-ray, radio..
  – As deep as the best instruments (2 years ago)
  – It is up when you are up.
    The “seeing” is always great
    (no working at night, no clouds no moons no..).
  – It’s a smart telescope:
    links objects and data to literature on them.
What’s needed?
(not drawn to scale)

Scientists
Science Data & Questions

Miners
Data Mining Algorithms

Plumbers
Database
To store data
Execute Queries

Tools
Question & Answer Visualization
Why Astronomy Data?

• It has no commercial value
  – No privacy concerns
  – Can freely share results with others
  – Great for experimenting with algorithms

• It is real and well documented
  – High-dimensional data (with confidence intervals)
  – Spatial data
  – Temporal data

• Many different instruments from many different places and many different times

• Federation is a goal

• The questions are interesting
  – How did the universe form?

• There is a lot of it (petabytes)
Why Astronomy Data?

• There is lots of it
  – High dimensional
  – Spatial
  – temporal

• Great sandbox for
data mining algorithms
  – Can share cross company
  – University researchers

• Great way to teach both
  Astronomy and
  Computational Science

• Want to federate many instruments
But…some science is hitting a wall. FTP and GREP are not adequate.

- You can GREP 1 MB in a second.
- You can GREP 1 GB in a minute.
- You can GREP 1 TB in 2 days.
- You can GREP 1 PB in 3 years.
- Oh!, and 1PB ~10,000 disks.

At some point you need **indices** to limit search and analysis, parallel data search and analysis.

- This is where databases can help.
SkyServer
SkyServer.SDSS.org

- Like the TerraServer, but looking the other way: a picture of ¼ of the universe
- Pixels + Data Mining
- Astronomers get about 400 attributes for each “object”
- Get Spectrograms for 1% of the objects
Goal: Easy Data Publication & Access

- Augment FTP with data query:
  Return intelligent data subsets
- Make it easy to
  - Publish: Record structured data
  - Find:
    - Find data anywhere in the network
    - Get the subset you need
  - Explore datasets interactively
- Realistic goal:
  - Make it as easy as publishing/reading web sites today.
Web Services: The Key?

• **Web SERVER:**
  - Given a url + parameters
  - Returns a web page (often dynamic)

• **Web SERVICE:**
  - Given a XML document (soap msg)
  - Returns an XML document
  - Tools make this look like an RPC.
    - $F(x,y,z)$ returns $(u, v, w)$
  - Distributed objects for the web.
  - + naming, discovery, security,..

• **Internet-scale distributed computing**
Data Federations of Web Services

• Massive datasets live near their owners:
  – Near the instrument’s software pipeline
  – Near the applications
  – Near data knowledge and curation
  – Super Computer centers become Super Data Centers

• Each Archive publishes a web service
  – Schema: documents the data
  – Methods on objects (queries)

• Scientists get “personalized” extracts

• Uniform access to multiple Archives
  – A common global schema
Grid and Web Services Synergy

- I believe the Grid will be many web services
- IETF standards Provide
  - Naming
  - Authorization / Security / Privacy
  - Distributed Objects
    Discovery, Definition, Invocation, Object Model
  - Higher level services: workflow, transactions, DB,..
SkyQuery (http://skyquery.net/)

- Distributed Query tool using a set of web services
- Feasibility study, built in 6 weeks from scratch
  - Tanu Malik (JHU CS grad student)
  - Tamas Budavari (JHU astro postdoc)
  - With help from Szalay, Thakar, Gray
- Implemented in C# and .NET
- Allows queries like:

```sql
SELECT o.objId, o.r, o.type, t.objId
FROM SDSS:PhotoPrimary o,
     TWOMASS:PhotoPrimary t
WHERE XMATCH(o,t)<3.5
  AND AREA(181.3,-0.76,6.5)
  AND o.type=3 and (o.I - t.m_j)>2
```
Structure

Web Page

SkyNode 2Mass

SkyNode First

Image cutout

SkyQuery

SkyNode SDSS
Show Cutout Web Service
Outline

• The World Wide Telescope Idea
• Data Mining the Sloan Digital Sky Survey
• Spherical Geometry in SQL
Working Cross-Culture
How to design the database: Scenario Design

• Astronomers proposed 20 questions
• Typical of things they want to do
• Each would require a week of programming in tcl / C++/ FTP
• Goal, make it easy to answer questions
• DB and tools design motivated by this goal
  – Implemented utility procedures
  – JHU Built Query GUI for Linux /Mac/.. clients
The 20 Queries

Q1: Find all galaxies without unsaturated pixels within 1' of a given point of ra=75.327, dec=21.023
Q2: Find all galaxies with blue surface brightness between 23 and 25 mag per square arcseconds, and -10<super galactic latitude (sgb) <10, and declination less than zero.
Q3: Find all galaxies brighter than magnitude 22, where the local extinction is >0.75.
Q4: Find galaxies with an isophotal surface brightness (SB) larger than 24 in the red band, with an ellipticity>0.5, and with the major axis of the ellipse having a declination of between 30'' and 60''arc seconds.
Q5: Find all galaxies with a deVaucouleurs profile (r^{1/4} falloff of intensity on disk) and the photometric colors consistent with an elliptical galaxy. The deVaucouleurs profile
Q6: Find galaxies that are blended with a star, output the deblended galaxy magnitudes.
Q7: Provide a list of star-like objects that are 1% rare.
Q8: Find all objects with unclassified spectra.
Q9: Find quasars with a line width >2000 km/s and 2.5<redshift<2.7.
Q10: Find galaxies with spectra that have an equivalent width in Ha >40Å (Ha is the main hydrogen spectral line.)

Q11: Find all elliptical galaxies with spectra that have an anomalous emission line.
Q12: Create a grided count of galaxies with u-g>1 and r<21.5 over 60<declination<70, and 200<right ascension<210, on a grid of 2', and create a map of masks over the same grid.
Q13: Create a count of galaxies for each of the HTM triangles which satisfy a certain color cut, like 0.7u-0.5g-0.2i<1.25 && r<21.75, output it in a form adequate for visualization.
Q14: Find stars with multiple measurements and have magnitude variations >0.1. Scan for stars that have a secondary object (observed at a different time) and compare their magnitudes.
Q15: Provide a list of moving objects consistent with an asteroid.
Q16: Find all objects similar to the colors of a quasar at 5.5<redshift<6.5.
Q17: Find binary stars where at least one of them has the colors of a white dwarf.
Q18: Find all objects within 30 arcseconds of one another that have very similar colors: that is where the color ratios u-g, g-r, r-I are less than 0.05m.
Q19: Find quasars with a broad absorption line in their spectra and at least one galaxy within 10 arcseconds. Return both the quasars and the galaxies.
Q20: For each galaxy in the BCG data set (brightest color galaxy), in 160<right ascension<170, -25<declination<35 count of galaxies within 30''of it that have a photoz within 0.05 of that galaxy.

Also some good queries at: http://www.sdss.jhu.edu/ScienceArchive/sxqt/sxQT/Example_Queries.html
Two kinds of SDSS data in an SQL DB
(objects and images all in DB)
• 100M Photo Objects ~ 400 attributes

400K Spectra
with
~30 lines/spectrum
An easy one: Q7:
Provide a list of star-like objects that are 1% rare.

• Found 14,681 buckets,
  first 140 buckets have 99%
  time 104 seconds
• Disk bound, reads 3 disks at 68 MBps.

Select cast((u-g) as int) as ug,
    cast((g-r) as int) as gr,
    cast((r-i) as int) as ri,
    cast((i-z) as int) as iz,
    count(*)           as Population
from stars
group by    cast((u-g) as int), cast((g-r) as int),
            cast((r-i) as int), cast((i-z) as int)
order by count(*)
An easy one Q15: Provide a list of moving objects consistent with an asteroid.

• Sounds hard but there are 5 pictures of the object at 5 different times (colors) and so can compute velocity.
• Image pipeline computes velocity.
• Computing it from the 5 color x,y would also be fast
• Finds 285 objects in 3 minutes, 140MBps.

```
select objectId,
       sqrt(power(rowv,2)+power(colv,2)) as velocity
from photoObj
where (power(rowv,2) + power(colv, 2)) between 50 and 1000
     -- return object ID
     -- check each object.
     -- square of velocity
     -- huge values =error
```
**Q15: Fast Moving Objects**

- **Find near earth asteroids:**

```
SELECT r.objID as rId, g.objId as gId, r.run, r.camcol, r.field as field, g.field as gField,
       r.ra as ra_r, r.dec as dec_r, g.ra as ra_g, g.dec as dec_g,
       sqrt( power(r.cx -g.cx,2)+ power(r.cy-g.cy,2)+power(r.cz-g.cz,2) )*(10800/PI()) as distance
FROM PhotoObj r, PhotoObj g
WHERE
  r.run = g.run and r.camcol=g.camcol and abs(g.field-r.field)<2  -- the match criteria
  -- the red selection criteria
  and ((power(r.q_r,2) + power(r.u_r,2)) > 0.111111 )
  and r.fiberMag_r between 6 and 22 and r.fiberMag_r < r.fiberMag_i
  and r.parentID=0 and r.fiberMag_r < r.fiberMag_u and r.fiberMag_r < r.fiberMag_z
  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 )
  and g.fiberMag_g between 6 and 22 and g.fiberMag_g < g.fiberMag_i
  and g.fiberMag_g < g.fiberMag_u 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 matchup of the pair
  and sqrt((power(r.cx -g.cx,2)+ power(r.cy-g.cy,2)+power(r.cz-g.cz,2)))*(10800/PI())< 4.0
  and abs(r.fiberMag_r-g.fiberMag_g)< 2.0
```

- **Finds 3 objects in 11 minutes**
  - (or 27 seconds with an index)

- **Ugly,**
  - but consider the alternatives (c programs an files and…)
SELECT r.objID as rId, g.objID as gId, r.run, r.camcol, r.field as field, g.field as gField,
    r.ra as ra_r, r.dec as dec_r, g.ra as ra_g, g.dec as dec_g,
    sqrt((power(r.cx - g.cx,2) + power(r.cy - g.cy,2)) + power(r.cz - g.cz,2)) * (10800/PI()) as distance
FROM PhotoObj r, PhotoObj g
WHERE r.run = g.run and r.camcol = g.camcol and abs(g.field - r.field) < 2 -- the match criteria
    -- the red selection criteria
    and ((power(r.q_r,2) + power(r.u_r,2)) > 0.111111)
    and r.fiberMag_r between 6 and 22 and r.fiberMag_r < r.fiberMag_g and r.fiberMag_r < r.fiberMag_i
    and r.parentID = 0 and r.fiberMag_r < r.fiberMag_u and r.fiberMag_r < r.fiberMag_z
    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)
    and g.fiberMag_g between 6 and 22 and g.fiberMag_g < g.fiberMag_r and g.fiberMag_g < g.fiberMag_i
    and g.fiberMag_g < g.fiberMag_u 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 matchup of the pair
    and sqrt((power(r.cx - g.cx,2) + power(r.cy - g.cy,2) + power(r.cz - g.cz,2)) * (10800/PI()) < 4.0
    and abs(r.fiberMag_r - g.fiberMag_g) < 2.0
**A Hard One**

**Q14: Find stars with multiple measurements that have magnitude variations >0.1.**

- This should work, but SQL Server does not allow table values to be piped to table-valued functions.

```
select S.object_ID, S1.object_ID -- return stars that
from    Stars S,
        getNearbyObjEq(s.ra, s.dec, 0.017) as N -- N within 1 arcsec (3 pixels)
of S. Stars S1
    where S.Object_ID < N.Object_ID -- S1 different from S == N
        and N.Type = dbo.PhotoType('Star') -- S1 is a star (an optimization)
        and N.object_ID = S1.Object_ID -- N == S1
        and (  abs(S.u-S1.u) > 0.1 -- one of the colors is different.
            or abs(S.g-S1.g) > 0.1
            or abs(S.r-S1.r) > 0.1
            or abs(S.i-S1.i) > 0.1
            or abs(S.z-S1.z) > 0.1
         )
order by S.object_ID, S1.object_ID -- group the answer by parent star.
```

Returns a table of nearby objects
A Hard one: Second Try: Q14

Find stars with multiple measurements that have magnitude variations >0.1.

- Write a program with a cursor, ran for 2 days

-- Table-valued function that returns the binary stars within a certain radius
-- of another (in arc-minutes) (typically 5 arc seconds).
-- Returns the ID pairs and the distance between them (in arcseconds).
create function BinaryStars(@MaxDistanceArcMins float)
returns @BinaryCandidatesTable table(
    S1_object_ID bigint not null, -- Star #1
    S2_object_ID bigint not null, -- Star #2
    distance_arcSec float) -- distance between them
as
begin
    declare @star_ID bigint, @binary_ID bigint;-- Star's ID and binary ID
    declare @ra float, @dec float; -- Star's position
    declare @u float, @g float, @r float, @i float,@z float; -- Star's colors
    declare @star_ID bigint, @binary_ID bigint;-- Star's ID and binary ID
    declare @ra float, @dec float; -- Star's position
    declare @u float, @g float, @r float, @i float,@z float; -- Star's colors

    -- Open a cursor over stars and get position and colors
    declare star_cursor cursor
    for select object_ID, ra, [dec], u, g, r, i, z from Stars;
    open star_cursor;
    while (1=1) -- for each star
        begin -- get its attributes
            fetch next from star_cursor into @star_ID, @ra, @dec, @u, @g, @r, @i, @z;
            if (@@fetch_status = -1) break; -- end if no more stars
            insert into @BinaryCandidatesTable
                select @star_ID, S1.object_ID, -- return stars pairs
                sqrt(N.DotProd)/PI()*10800 -- and distance in arc-seconds
                from getNearbyObjEq(@ra, @dec, -- Find objects nearby S.
                    @MaxDistanceArcMins) as N, -- call them N.
                    Stars as S1 -- S1 gets N's color values
                where @star_ID < N.Object_ID -- S1 different from S
                    and N.objType = dbo.PhotoType('Star') -- S1 is a star
                    and N.object_ID = S1.object_ID -- join stars to get colors of S1==N
                    and (abs(@u-S1.u) > 0.1 -- one of the colors is different.
                        or abs(@g-S1.g) > 0.1
                        or abs(@r-S1.r) > 0.1
                        or abs(@i-S1.i) > 0.1
                        or abs(@z-S1.z) > 0.1)
        end;
        close star_cursor; -- end of loop over all stars
    deallocate star_cursor;
    return;
end
GO
select * from dbo.BinaryStars(.05)
A Hard one: Third Try

Q14: Find stars with multiple measurements that have magnitude variations >0.1.

- Use pre-computed neighbors table.
- Ran in 17 minutes, found 31k pairs.

```
-- Plan 2: Use the precomputed neighbors table
select top 100 S.object_ID, S1.object_ID, -- return star pairs and distance
       str(N.Distance_mins * 60,6,1) as DistArcSec
from Stars S,
     Neighbors N, -- N within 3 arcsec (10 pixels) of S.
     Stars S1 -- S1 == N has the color attributes
where S.Object_ID = N.Object_ID -- connect S and N.
    and S.Object_ID < N.Neighbor_Object_ID -- S1 different from S
    and N.Neighbor_objType = dbo.PhotoType('Star') -- S1 is a star (an optimization)
    and N.Distance_mins < .05 -- the 3 arcsecond test
    and N.Neighbor_object_ID = S1.Object_ID -- N == S1
    and (   abs(S.u-S1.u) > 0.1 -- one of the colors is different.
           or abs(S.g-S1.g) > 0.1
           or abs(S.r-S1.r) > 0.1
           or abs(S.i-S1.i) > 0.1
           or abs(S.z-S1.z) > 0.1
    )
-- Found 31,355 pairs (out of 4.4 m stars) in 17 min 14 sec.
```
The Pain of Going Outside SQL
(its fortunate that all the queries are single statements)

- Count parent objects
- 503 seconds for 14.7 M objects in 33.3 GB
- 66 MBps
- IO bound (30% of one cpu)
- 100 k records/cpu

- Use a cursor
- No cpu parallelism
- CPU bound
- 6 MBps, 2.7 k rps
- 5,450 seconds (10x slower)

```sql
select count(*)
from sxPhotoObj
where nChild > 0
```
```sql
declare @count int;
declare @sum int;
set @sum = 0;
declare PhotoCursor cursor for select
nChild from sxPhotoObj;
open PhotoCursor;
while (1=1)
    begin
        fetch next from PhotoCursor into @count;
        if (@@fetch_status = -1) break;
        set @sum = @sum + @count;
    end
close PhotoCursor;
deallocate PhotoCursor;
print 'Sum is: '+cast(@sum as varchar(12))
```
-- Query 19: Find quasars with a broad absorption line in their spectra
-- and at least one galaxy within 10 arcseconds.
-- Return both the quasars and the galaxies.

```sql
select Q.ObjID as Quasar_candidate_ID, G.ObjID as Galaxy_ID
into ##results
from PrimaryObjects as Q, Neighbors as N, Galaxies as G, SpecObj as S, SpecClass as SC, SpecLine as L, SpecLineNames as LN
where Q.ObjID = S.ObjID
  and S.SpecClass = SC.class
  and SC.name in ('QSO', 'HIZ_QSO')
  and S.SpecObjID = L.SpecObjID
  and L.LineID = LN.LineID
  and LN.Name != 'UNKNOWN'
  and L.ew < -10
  and Q.ObjID = N.ObjID
  and G.ObjID = N.NeighborObjID
  and N.NeighborObjType = dbo.fPhotoType('Galaxy')
  and N.distanceMins < 10/60
-- and it is within 10 arcseconds of the Q.
```
-- Query 19: Find quasars with a broad absorption line in their spectra
-- and at least one galaxy within 10 arcseconds.
-- Return both the quasars and the galaxies.

```sql
SELECT Q.ObjID AS Quasar_candidate_ID, G.ObjID AS Galaxy_ID
INTO #results
FROM PrimaryObjects Q, Galaxies G
WHERE Q.ID INSIDE G.ID 0.0
```

Query 1: Query cost (relative to the batch): 100.00
Query text: select Q.ObjID as Quasar_candidate_ID, G.ObjID as Galaxy_ID into #results from PrimaryObjects as Q, -- Q is the QIO candidate, Neighbors as N, -- N is the N...
Q15: Fast Moving Objects

- Find near earth asteroids:

```sql
SELECT r.objID as rId, g.objId as gId,
       dbo.fGetUrlEq(g.ra, g.dec) as url
FROM PhotoObj r, PhotoObj g
WHERE  r.run = g.run and r.camcol=g.camcol
       and abs(g.field-r.field)<2  -- nearby
       -- the red selection criteria
       and ((power(r.q_r,2) + power(r.u_r,2)) > 0.111111 )
       and r.fiberMag_r between 6 and 22 and r.fiberMag_r < r.fiberMag_i
       and r.parentID=0 and r.fiberMag_r < r.fiberMag_u
       and r.fiberMag_r < r.fiberMag_z
       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 )
       and g.fiberMag_g between 6 and 22 and g.fiberMag_g < g.fiberMag_i
       and g.fiberMag_g < g.fiberMag_u 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 matchup of the pair
       and sqrt(power(r.cx -g.cx,2)+ power(r.cy-g.cy,2)+power(r.cz-g.cz,2))
       and abs(r.fiberMag_r-g.fiberMag_g)< 2.0
```

- Finds 3 objects in 11 minutes
  – (or 52 seconds with an index)
- Ugly, but consider the alternatives (c programs and files and time... )
Performance (on current SDSS data)

- Run times: on 15k$ HP Server (2 cpu, 1 GB, 8 disk)
- Some take 10 minutes
- Some take 1 minute
- Median ~ 22 sec.
- Ghz processors are fast!
  - (10 mips/IO, 200 ins/byte)
  - 2.5 m rec/s/cpu
Outline

• The World Wide Telescope Idea
• Data Mining the Sloan Digital Sky Survey
• Spherical Geometry in SQL
Spherical Geometry

• Astronomy has redshifts (3D), but often works with celestial sphere (2D)
• Distance ~ arc angle
• Extended SQL to have “neighbor functions”
  – GetNearestObject() returns a table with one row
  – GetNearbyObjects() returns a table
Hierarchical Triangular Mesh (HTM)

Szalay, Kunszt, Brunner http://www.sdss.jhu.edu/htm

• Every object has a 20-deep Mesh ID
• Given an area routine returns set of covering triangles
• Each triangle implies range query: htmID in triangle iff htmID in [triangle.min…triangle.max)
• Reject false positives with careful geometry test
• Very fast: 10,000 triangles / second / cpu
Using Hierarchical Triangular Mesh

• select *
  from photoObj as p,
    fHtmCover(x,y,z,r) as n
  where p.htmID between n.start and n.end
  and (2*asin(sqrt(power(x-cx,2)+power(y-cy,2)+power(z-cz,2))/2)) < radians(r)

• This is packaged as:
  – fGetNearbyObjects(x,y,z,r)
Spherical Areas

- $\text{SphericalArea} = \{ \pm \text{ConvexArea} \}$
- $\text{ConvexArea} = \{ \pm \text{SphericalEdge} \} = \{ \text{PlaneSphereIntersect} \}$
- $\text{Plane} =$ normal unit vector $v$ $(vx, vy, vz)$ length $l$.
- Point $p = (x, y, z)$ on the unit sphere is “inside” the edge if $(xyz) \cdot (vx, vy, vz) > l$.
- A point is inside a convex area if it is inside each of the edges.
- Non-convex areas are convex area unions.
- Swiss-cheese areas (holes in them) are positive and negative convex areas actually, just negative lengths.
Areas as Tables

--- An area is a set of convexes that have a set of edges.

```sql
create table Area (  
  AreaID integer, -- the unique identifier of the area  
  ConvexID integer, -- the unique identifier of a convex  
  EdgeID integer, -- unique id of the edge of an edge  
  x float, -- the xyz vector of the edge (v)  
  y float, --  
  z float, --  
  l float, -- the vector length.  
primary key (AreaID, ConvexID, EdgeID)  
)
```
Point in Convex

```sql
declare @aID int -- area ID is a parameter

select * -- return all points
from Points p
where not exists ( -- where there is no “outside” edge.
    select EdgeID -- for all edges
    from Area a
    where AreaID = @aID -- in the area
        and (p.x*a.x + p.y*a.y + p.z*a.z) < a.l) -- test “outside”
)
```
Point in Polygon
(union of convexes)

```
select *
from Points p
where exists (  
    select ConvexID  
    from area a  
    where AreaID = @aID  
    and (p. x*a.x + p. y*a.y + p. z*a.z) < a.l  
    group by all ConvexID  
    having count(*) = 0
)
```
A Simple Extension

- Do this for the plane and for N-space (rather than the sphere)
- That’s easy, so ….
  A harder problem that took me 2 years,
  So, you should get it in an hour...😊
A harder problem:  
compute the materialized view:  
(object, neighborObject, distance)  
for all distance less than 30 asec

• Using nearby function: 1 cpu day/ 1 Mobj
• Using set operators: 1 cpu day/ 100 Mobj
• An interesting algorithm:
  – hint: break into horizontal zones  
  30asec high
  – Join 3 pairs of zones.
  – Worry about wrap-around on the sphere
So what?

• SQL is a functional programming language, perhaps the most popular one.
• Set problems are easy in set-oriented languages.
• We (I) have not been thinking in sets (since I left the math department).
• This set-oriented approach is faster than the HTM function because it is inside the DB and it is batched
Summary

• The World Wide Telescope Idea
• Data Mining the Sloan Digital Sky Survey
• Spherical Geometry in SQL
Call to Action

• If you do data visualization: we need you (and we know it).
• If you do databases: here is some data you can practice on.
• If you do distributed systems: here is a federation you can practice on.
• If you do data mining here is a dataset to test your algorithms.
• If you do astronomy educational outreach here is a tool for you.
SkyServer references

http://SkyServer.SDSS.org/
http://research.microsoft.com/pubs/
http://research.microsoft.com/Gray/SDSS/ (download personal SkyServer)

• Data Mining the SDSS SkyServer Database
  Jim Gray; Peter Kunszt; Donald Slutz; Alex Szalay; Ani Thakar; Jan Vandenberg; Chris Stoughton Jan. 2002 40 p.
  An earlier paper described the Sloan Digital Sky Survey’s (SDSS) data management needs [Szalay1] by defining twenty database queries and twelve data visualization tasks that a good data management system should support. We built a database and interfaces to support both the query load and also a website for ad-hoc access. This paper reports on the database design, describes the data loading pipeline, and reports on the query implementation and performance. The queries typically translated to a single SQL statement. Most queries run in less than 20 seconds, allowing scientists to interactively explore the database. This paper is an in-depth tour of those queries. Readers should first have studied the companion overview paper "The SDSS SkyServer – Public Access to the Sloan Digital Sky Server Data" [Szalay2].

• SDSS SkyServer–Public Access to Sloan Digital Sky Server Data
  Jim Gray; Alexander Szalay; Ani Thakar; Peter Z. Zunszt; Tanu Malik; Jordan Raddick; Christopher Stoughton; Jan Vandenberg November 2001 11 p.: Word 1.46 Mbytes PDF 456 Kbytes
  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.

• The World-Wide Telescope
  Jim Gray; Alexander Szalay August 2001 6 p.: Word 684 Kbytes PDF 84 Kbytes
  All astronomy data and literature will soon be online and accessible via the Internet. The community is building the Virtual Observatory, an organization of this worldwide data into a coherent whole that can be accessed by anyone, in any form, from anywhere. The resulting system will dramatically improve our ability to do multi-spectral and temporal studies that integrate data from multiple instruments. The virtual observatory data also provides a wonderful base for teaching astronomy, scientific discovery, and computational science.

• Designing and Mining Multi-Terabyte Astronomy Archives
  Robert J. Brunner; Jim Gray; Peter Kunszt; Donald Slutz; Alexander S. Szalay; Ani Thakar June 1999 8 p.: Word (448 Kybytes) PDF (391 Kbytes)
  The next-generation astronomy digital archives will cover most of the sky at fine resolution in many wavelengths, from X-rays, through ultraviolet, optical, and infrared. The archives will be stored at diverse geographical locations. One of the first of these projects, the Sloan Digital Sky Survey (SDSS) is creating a 5-wavelength catalog over 10,000 square degrees of the sky (see http://www.sdss.org/). The 200 million objects in the multi-terabyte database will have mostly numerical attributes in a 100+ dimensional space. Points in this space have highly correlated distributions.
  The archive will enable astronomers to explore the data interactively. Data access will be aided by multidimensional spatial and attribute indices. The data will be partitioned in many ways. Small tag objects consisting of the most popular attributes will accelerate frequent searches. Splitting the data among multiple servers will allow parallel, scalable I/O and parallel data analysis. Hashing techniques will allow efficient clustering, and pair-wise comparison algorithms that should parallelize nicely. Randomly sampled subsets will allow de-bugging otherwise large queries at the desktop. Central servers will operate a data pump to support sweep searches touching most of the data. The anticipated queries will require special operators related to angular distances and complex similarity tests of object properties, like shapes, colors, velocity vectors, or temporal behaviors. These issues pose interesting data management challenges.