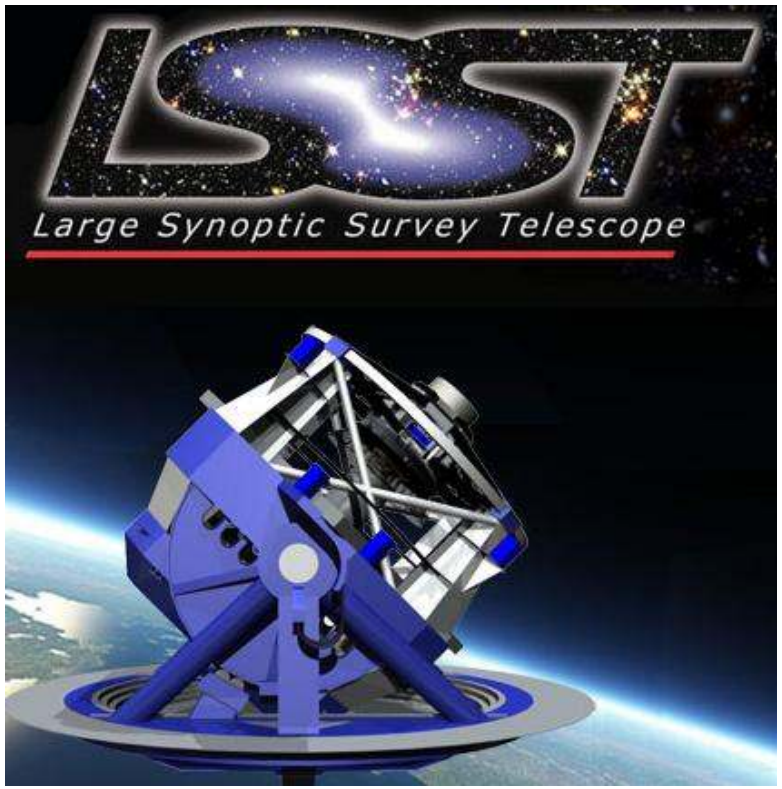
SciDB: A DBMS for Scientific Applications

Michael Stonebraker

Outline

- Origins of SciDB
- Requirements
- What it looks like
- The future



O(100) petabytes

QuickTime™ and a
decompressor
are needed to see this picture.

QuickTime™ and a
decompressor
are needed to see this picture.

LSST Data (100 Pbytes)

- Raw imagery
  - 2D arrays of telescope readings
- "Cooked" into observations
  - Image intensity algorithm (data clustering)
  - Spatial data
- Further cooked into "trajectories"
  - Similarity query
  - Constrained by maximum distance

Example LSST Queries

- Re-cook raw imagery with “my algorithm”
- Find all observations in a spatial region
- Find all trajectories that intersect a cylinder in time

LSST Data Management Team (Becla, Lim)

- Organized 1st XLDB workshop
- In October 2007
- Because they had no idea how to do LSST data management

1st XLDB

- Present were
  - Science guys (who all said “I am in trouble and RDBMS is a non starter”)
  - Vendors (who said RDBMSs were terrific)
  - Big web guys (who were all rolling their own)
  - Dewitt and I (who said we would like to help)

Result

- A workshop in early 2008
- A set of use cases
- A set of requirements
  - Data management
  - Complex analytics **IN THE SAME SYSTEM**
  - Scalability
  - **Commercial quality**
  - **Open source**
  - Provenance, uncertainty, …

Result

- An LSST-style benchmark
  - Paper available now
  - Data/code in a couple of months
  - SciDB is X100 sharded MySQL

Since 2008

- We have researched architecture and API issues
- We have organized a community of experienced and motivated volunteers
- We have raised venture capital and NSF funding
- Released 3 versions of the code

Who is "We"

- CEO, me, engineering director, four engineers in Waltham, Ma.
- 4 engineers and 2 math wizards in Russia
- 2 QA people in India
- Chief Plumber/Yak herder in California
- Science advisors from several disciplines

Who is "We"

- Stan Zdonik, Dave Maier, Sam Madden, Magda Balazinka, Ugur Cetintemal, Jignesh Patel, and a bunch of students working on architecture and API issues (with NSF support)
- Volunteers from SLAC (LSST) and elsewhere helping out
- Your name could be here!!!!
- See SciDB.org for more info

SciDB Mission

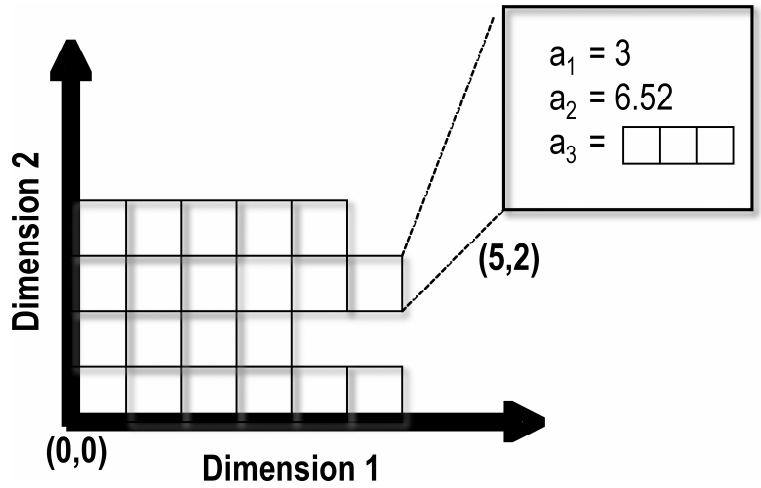
- Build what the science guys want
- Sell it to Big Pharma, Insurance and others with machine generated data to analyze
  - to pay the bills
  -

What is SciDB?

- Data model
  - Arrays, not tables
  - Much data is naturally an array
  - Stat is almost all array-based
- Storage architecture
  - N-dimensional chunking on disk
  - Efficient to go along any axis
- Query language
  - Make it look like SQL
- Other stuff

Data Model

- Multi-dimensional arrays
  - Cells can be tuples
  - Nesting will come later
- Time is an automatically supported extra dimension
- Ragged arrays allow each row/column to have a different dimensionality
- Support for multiple flavors of 'null'
  - Data missing, data coming, ….



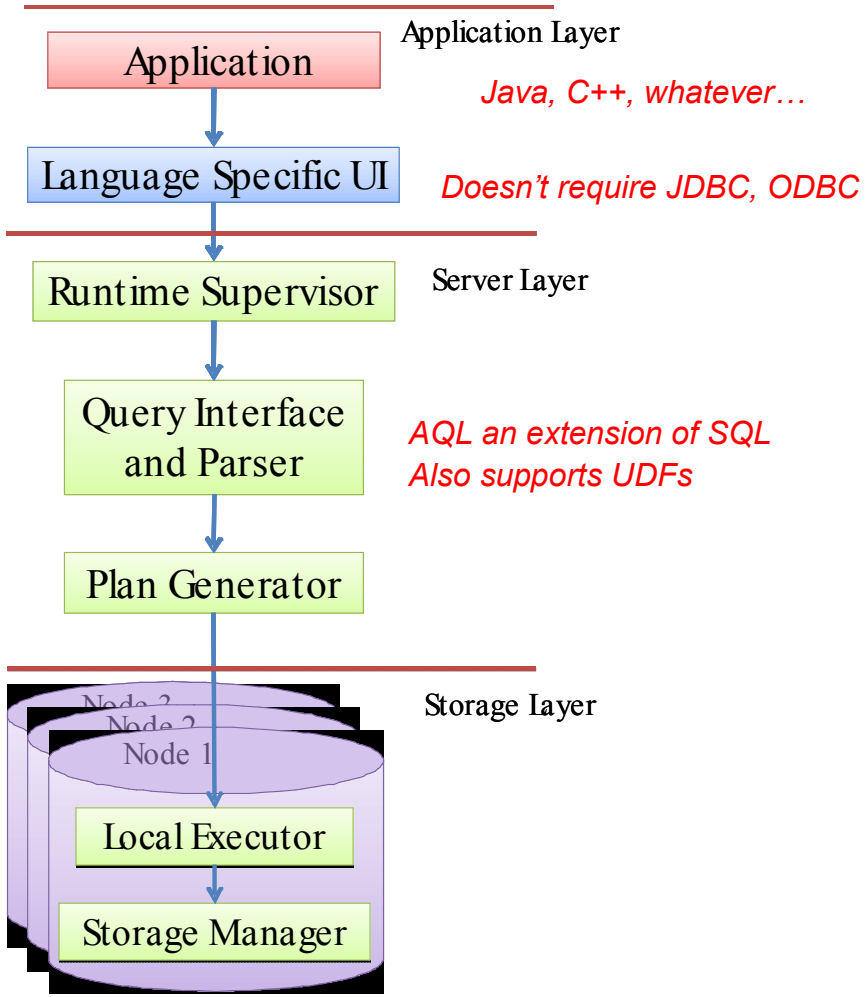
Array-oriented storage manager

- Optimized for both dense and sparse array data
  - Different data storage, compression, and access
- Arrays are “chunked”
  - in multiple dimensions
- Chunks are partitioned across a collection of nodes
- Chunks have ‘overlap’
  - to support neighborhood operations
- No overwrite

QuickTime™ and a
decompressor
are needed to see this picture.

Architecture

- Shared nothing cluster
  - 10's–1000's of nodes
  - Commodity hardware
  - TCP/IP between nodes
  - Linear scale-up
- Each node has a processor and storage
- Queries refer to arrays as if not distributed
- Query planner optimizes queries for efficient data access & processing
- Query plan runs on a node's local executor & storage manager



SciDB DDL

```
CREATE ARRAY Test_Array
    < A: integer NULLS,
      B: double,
      C: USER_DEFINED_TYPE >
    [I=0:99999,1000, 10, J=0:99999,1000, 10 ]
```

attribute names	dimension names	chunk size	overlap
A, B, C	I, J	1000	10

Array Query Language (AQL)

- Array data management
  - e.g. filter, aggregate, join, etc.
- Statistical & linear algebra operations
  - multiply, QR factorization, etc.
  - parallel, disk-oriented
- User-defined types and functions (Postgres-style)
  - Scalar functions
  - Aggregates
  - Array functions (e.g. Multiply)

Array Query Language (AQL)

```
SELECT Geo-Mean ( T.B )
FROM Test_Array T
WHERE
   T.I BETWEEN :C1 AND :C2
AND T.J BETWEEN :C3 AND :C4
AND Foo (T.C) > .75
GROUP BY T.I;
```

User-defined aggregate on an attribute B in array T

Subsample

Filter

Group-by

So far as SELECT / FROM / WHERE / GROUP BY queries are concerned, there is little logical difference between AQL and SQL

Matrix Multiply

```
CREATE ARRAY TS_Data < A1:int32, B1:double >
                                [ I=0:99999,1000,0, J=0:3999,100,0 ]

multiply (project (TS_Data, B1),
          transpose(project (TS_Data, B1)))
```

100K x 4K

Project on one attribute

- Illustrates AFL– cascaded operators in a functional language
- AQL (will be) compiled into AFL

Other Features Science Guys Want
(These could be in an RDBMS, but aren't)

- Uncertainty (normal distribution)
  - Data has error bars
  - Which must be carried along in the computation
    - interval arithmentic

Other Features

- Time travel
  - Don’t fix errors by overwrite
  - i.e. keep all the data – in the extra time dimension
- Named versions (coming)
  - Recalibration usually handled this way

Other Features

- Provenance (lineage) (Eugene Wu)
  - Walk backwards from a cell value
  - To find source of error
  - Walk forward from bad data
  - To fix it
  - As efficiently as possible with a space bound

Current Performance of SciDB

- Performance on stat operations
  - Comparable to or modestly beats R, but degrades in a similar way when data does not fit in main memory
  - But multi-node parallelism provided -- much better scalability
- Performance on SQL operations
  - Generally better-to-much-better than Postgres, especially on multi-attribute restrictions
  - And multi-node parallelism provided
- And we can do stuff they can't….
- Don't ever use SciDB for a transactional workload!!!
- Next release (November) will vectorize operations
  - and go a factor of 3-5 faster

So What Do We Need From You?

- Try out SciDB and let us know what you think
  - Unless you tell us what you want, we have no chance of doing it
- What stat operations do you want?
  - With what semantics
  - And how to parallelize them?
  - And how to deal with dense vs sparse data
  - And what to do with big (disk) data
  - And please volunteer to write them as UDF libraries!!
- What exactly do you want provenance to do?

What Keeps Me Up at Night?

- Storage Architecture
  - Big chunks (Mbytes) to optimize disk reads
  - Small tiles (Kbytes) to optimize L2 cache
  - But we should probably move to a multi-level scheme (Balazinska)
  - How to do compression and multiple nulls (Cetintemal)
- Who’s going to write 500 stat functions?
  - And maintain them

What Keeps Me Up at Night?

- How to interface to R (for Vis if nothing else)(Maier)
- How to deal with "in situ" data
  - E.g. HDF5 (SLAC)
- How to construct a clean array query language
  - Semantics of ragged arrays
  - Semantics of multiple kinds of nulls
  - Semantics of redimension
  - Tension between AQL and AFL

The Future

- Speed (next release)
- HA
- Reprovisioning on the fly
- No knobs/ automatic DB designer
- Domain-specific UDF libraries
- Your good idea goes here!!!!

Summary

- Current system is workable
- Next February we will have something that is production-ready:
  - Feature complete
  - Fast
  - Well documented