SciDB
An Open Source Data Base Project
Dong Liang
Zhao Xuting

Outline
♦ What is SciDB
♦ Data Model
♦ Query language
♦ Operations
♦ Storage and partition
♦ Other features
♦ Discussions

Why SciDB
♦ One thing does not fit all
♦ Tables?
  ♦ Makes a few of you happy
♦ But
  ♦ PanStarrs (Alex Szalay) wants arrays and scalability
♦ Nested multidimensional arrays
♦ Array values are a tuple of values and arrays

Data Model
♦ Basic Arrays:
  ♦ Basic syntax:
    ♦ define ArrayType ((name = Type-1)) ((dname))
    ♦ define Remote (s1 = float, s2 = float, s3 = float) (I, J)
  ♦ Positive integer dimensions, no gaps
  ♦ Bounded or unbounded
    ♦ create My_remote as Remote [1024, 1024]
    ♦ create My_remote_2 as Remote [*, *]

Data Model
♦ Enhanced Arrays
  ♦ Irregular boundary
  ♦ Coordinate systems
  ♦ User defined functions that map integers to something else
  ♦ E.g. Enhance A with Scale10
  ♦ Use dimension notation to access, e.g.
    ♦ A [7, 8] or A[I = 7, J = 8]
    ♦ A(20, 50) or A (K = 20, L = 50).

SciDB Query Language
♦ “Parse-tree” representation of array operations
♦ With a “binding” to:
  ♦ MatLab
  ♦ C++
  ♦ Python
  ♦ IDL
  ♦ There may be more....
♦ User defined operations and data types (Postgres-style)
SciDB Operators

- Structural-operators
  - Sjoin
  - Reshape

- Data-operators
  - filter: "select" function
  - Aggregate: aggregating a sub-dimensional-array
  - Cjoin

Data Storage and Partition

- Data Storage Model
  - Arrays are “chunked” in storage
    - Chunk size can vary or be constant
  - Chunks are partitioned across the grid
  - Go for scalability to petabytes

- With built-in high availability and failover
- And built in disaster recovery

In Situ Processing

- Operate on data with loading it
- Supported by a SciDB self-describing file format
  - And some number of adaptors, e.g. HDF-5, NetCDF
  - Or write your own

Other Features

- Uncertainty
  - Data has error bars
  - Which must be carried along in the computation (interval arithmetic)
  - Will look at more sophisticated error models later

- Provenance
  - What calibration generated the data
  - What was the “cooking” algorithm
  - In general – repeatability of data derivation
  - Supported by a command log
    - with query facilities (interesting research problem)
  - And redo

Other Features

- Time travel
  - Don’t fix errors by overwrite
  - I.e. keep all of the data
  - Supported by an extra array dimension (history)

- Named versions
  - Recalibration usually handled this way
  - Supported by allocating an array for the new version and “diffing” against its parent
Some discussions

- Does SciDB have a good chance at success?
- How to map time points with the integer dimension values in a “history dimension”?
- Is “history dimension” the proper way for time travel?
- How to realize the provenance, uncertainty, and optimization of SciDB?
- Possible performance testing

- Thanks!
- Any questions?