SciDB

Group 19:

Smitha Malur Muralidhar
Purva Kolhatkar
Manasi Pradhan
Matthew Tschiggfrie

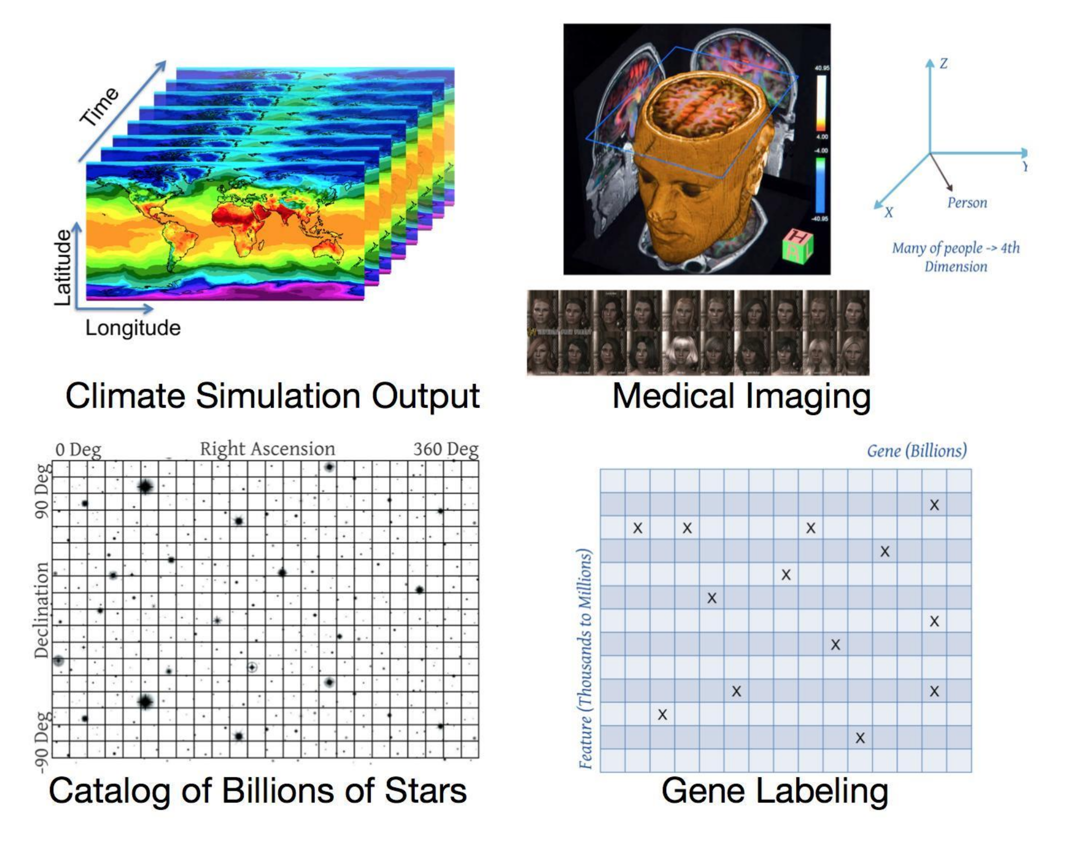
Outline

1. Why SciDB?
2. SciDB Architecture
3. SciDB-Py
4. SciDB-R
5. Popular Applications
6. Advantages and Disadvantages

What does scientific data look like?

- Extensive use of sensor arrays
- Scientific analysis involves sophisticated data processing.
- Data is large and is reused.

Why sciDB ?



Time
Latitude
Longitude
Climate Simulation Output
Z
Y
X
Person
Many of people -> 4th Dimension
Medical Imaging
0 Deg
Right Ascension
360 Deg
90 Deg
Declination
-90 Deg
Catalog of Billions of Stars
Gene (Billions)
Feature (Thousands to Millions)
Gene Labeling

Why sciDB ?

- Inadequacy of current commercial DBMS
- Custom database for every project.
- Natural relational table model doesn’t suit scientific data.
- Science community was reluctant to learn new programming language.

Who developed SciDB ?

2008 : Multi-institution project.

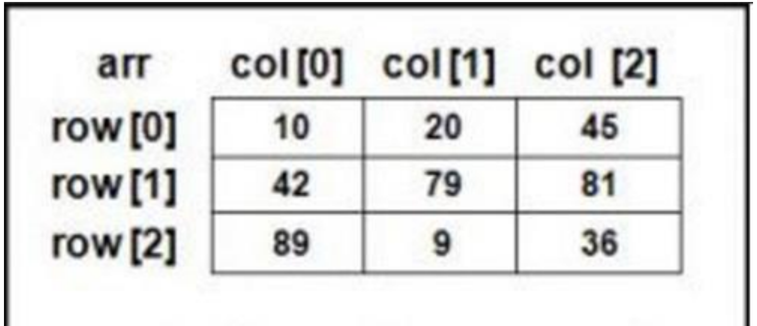
2011: Start-up Paradigm4 led by Michael stonebraker and Marilyn Matz.

What is sciDB ?

- ➔ Open source
- ➔ Distributed array database
- ➔ Horizontally scalable
- ➔ In database math
- ➔ ACID
- ➔ Integrated with R and python

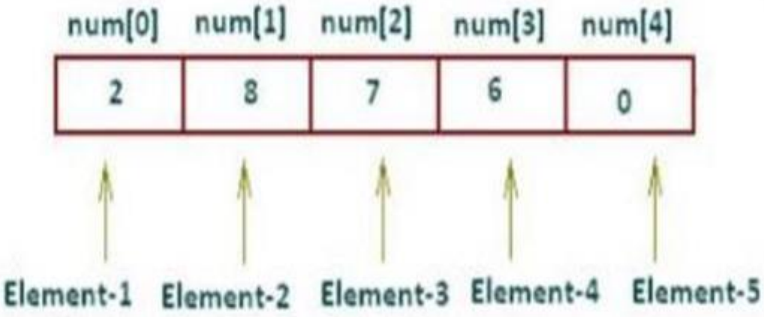
SciDB Architecture

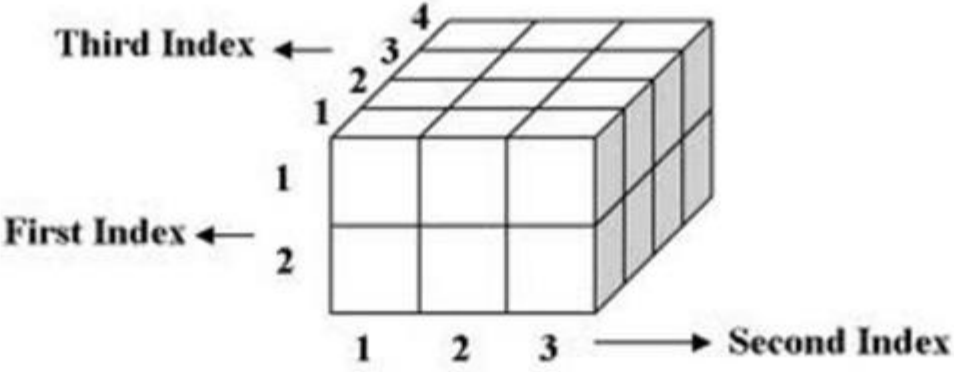
Arrays



arr	col [0]	col [1]	col [2]
row [0]	10	20	45
row [1]	42	79	81
row [2]	89	9	36

2D Array Arrangement





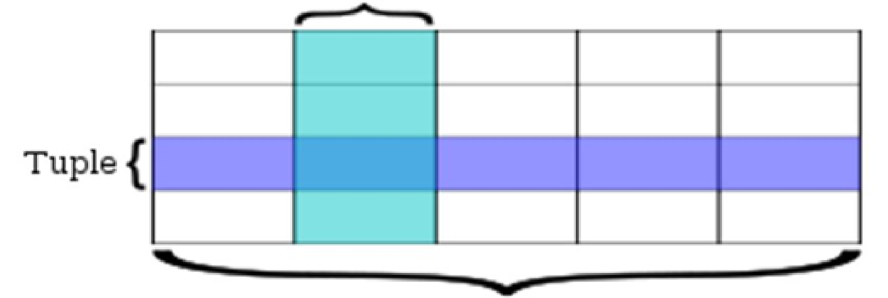
Three-dimensional array with twenty four elements

Array Data Model: Terminology Used

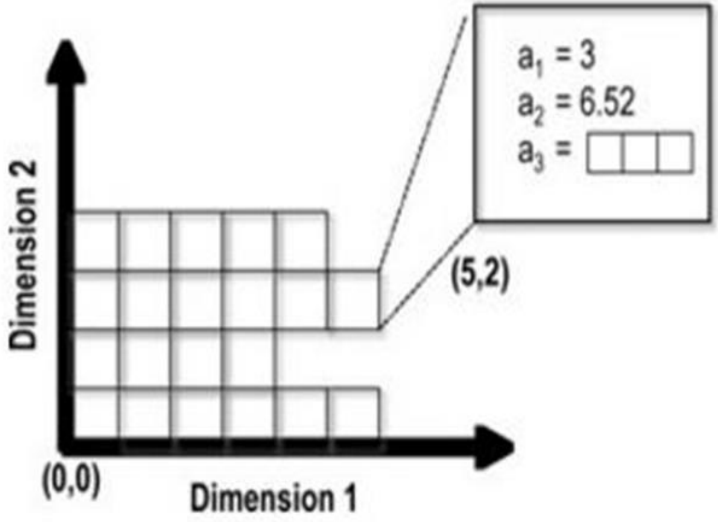
- Attributes

Price	Volume	Symbol	usec
450.61	150	“AAPL”	36013008713

- Table in Relational DBMS



- Dimensions



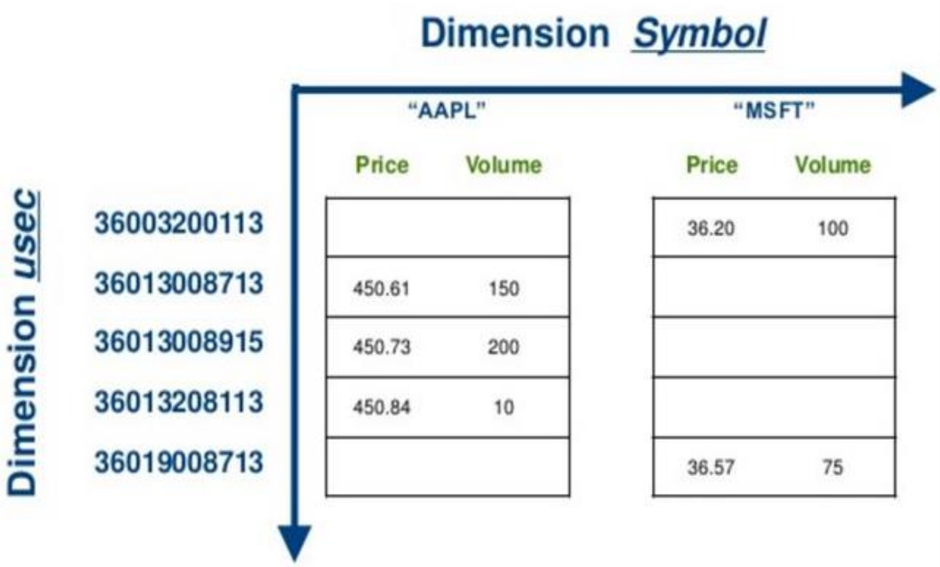
Array Data Model

	1996	2000	2004	2008
Dash	(Bailey , 9.84)	(Greene , 9.87)	(Gatlin , 9.85)	(Bolt , 9.69)
Steeplechase	(Keter , 487.12)	(Kosgei , 508.17)	(Kemboi , 485.81)	(Kipruto , 490.34)
Marathon	(Thugwane , 7956)	(Abera , 7811)	(Baldini , 7855)	(Wanjiru, , 7596)

Re-dimensioning arrays

Attributes

Dimension i	Price	Volume	Symbol	usec
1	450.61	150	"AAPL"	36013008713
2	450.73	200	"AAPL"	36013008915
3	450.84	10	"AAPL"	36013208113
4	36.57	75	"MSFT"	36019008713
5	36.20	100	"MSFT"	36003200113



Examples

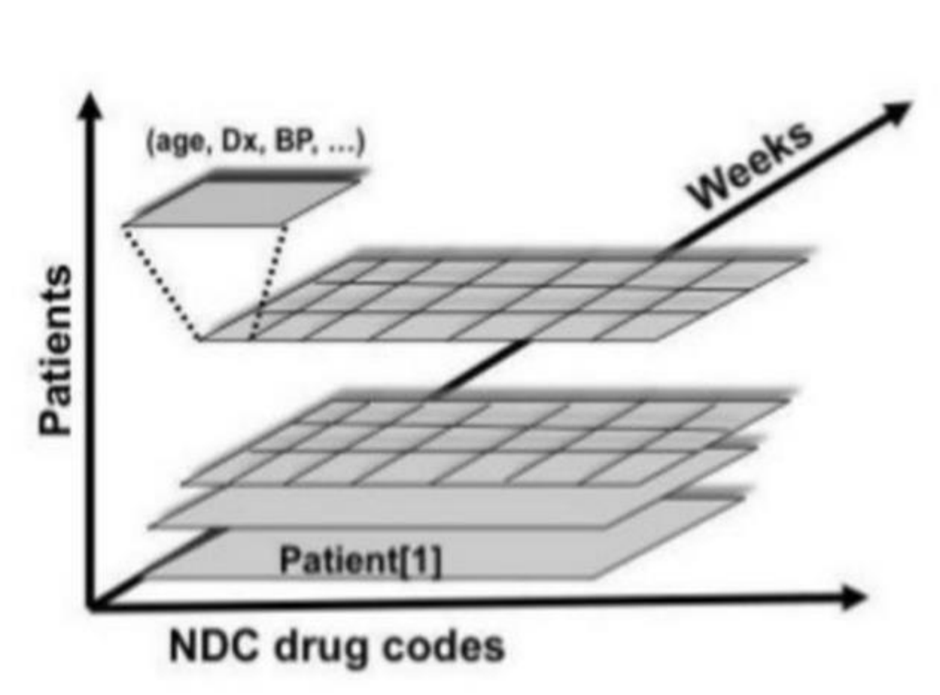


Figure 1

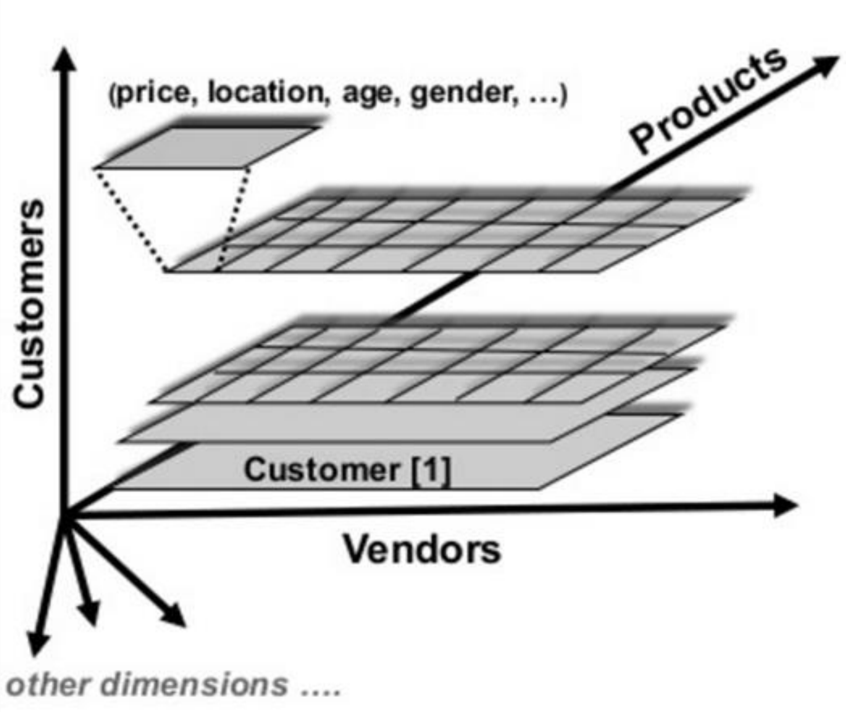


Figure 2

Range selection in Relational database

Relational Database

I	J	value
0	0	32.5
1	0	90.9
2	0	42.1
3	0	96.7
0	1	46.3
1	1	35.4
2	1	35.7
3	1	41.3
0	2	81.7
1	2	35.9
2	2	35.3
3	2	89.9
0	3	53.6
1	3	86.3
2	3	45.9
3	3	27.6

48 cells

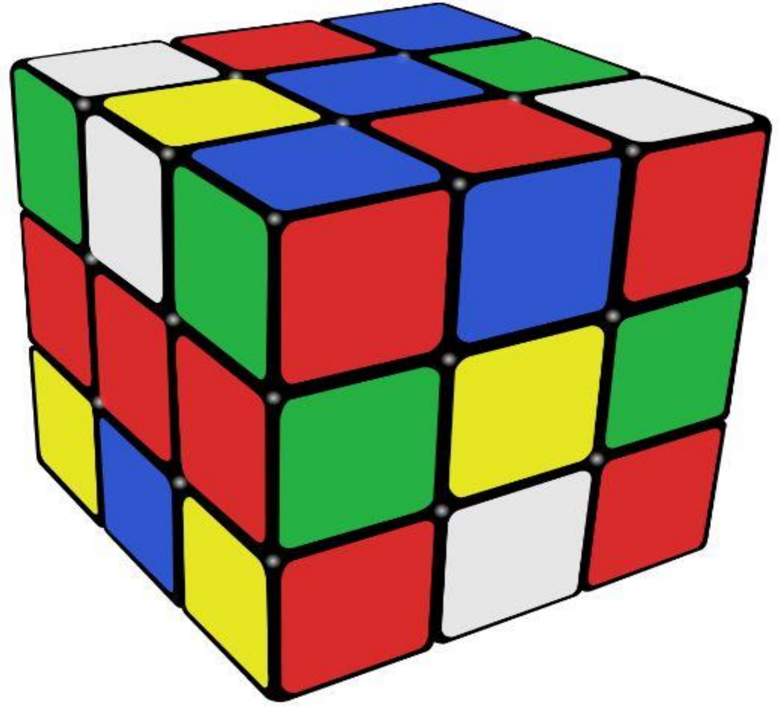
Array Database

32.5	46.3	81.7	53.6
90.9	35.4	35.9	86.3
42.1	35.7	35.3	45.9
96.7	41.3	89.9	27.6

16 cells

Range selection in SciDB

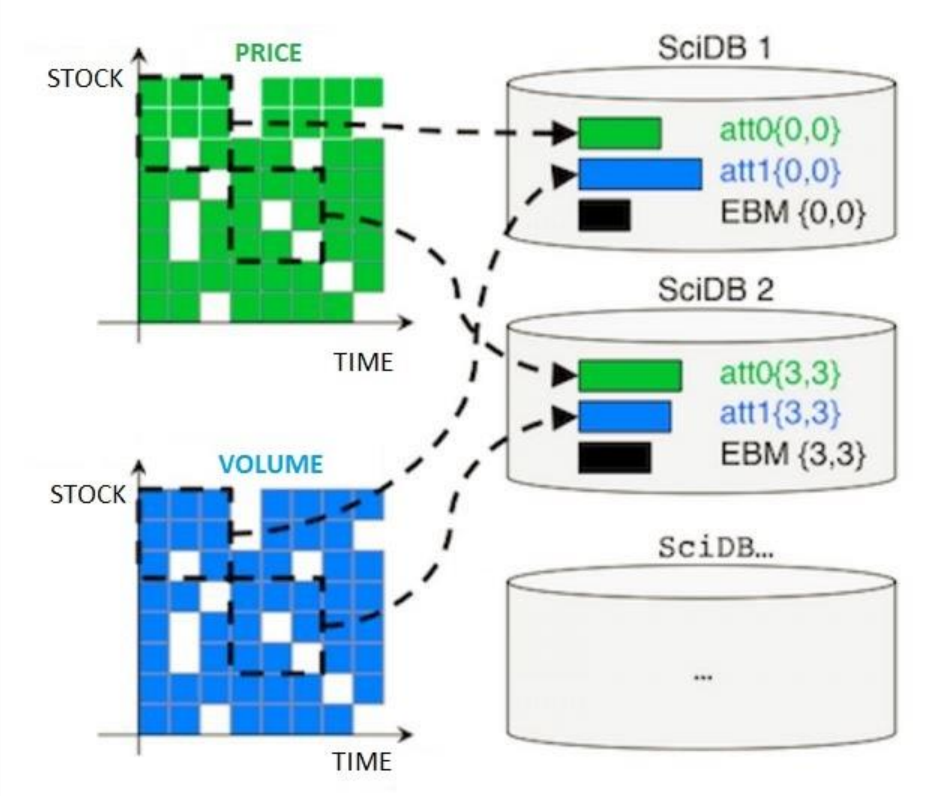
SciDB chunks



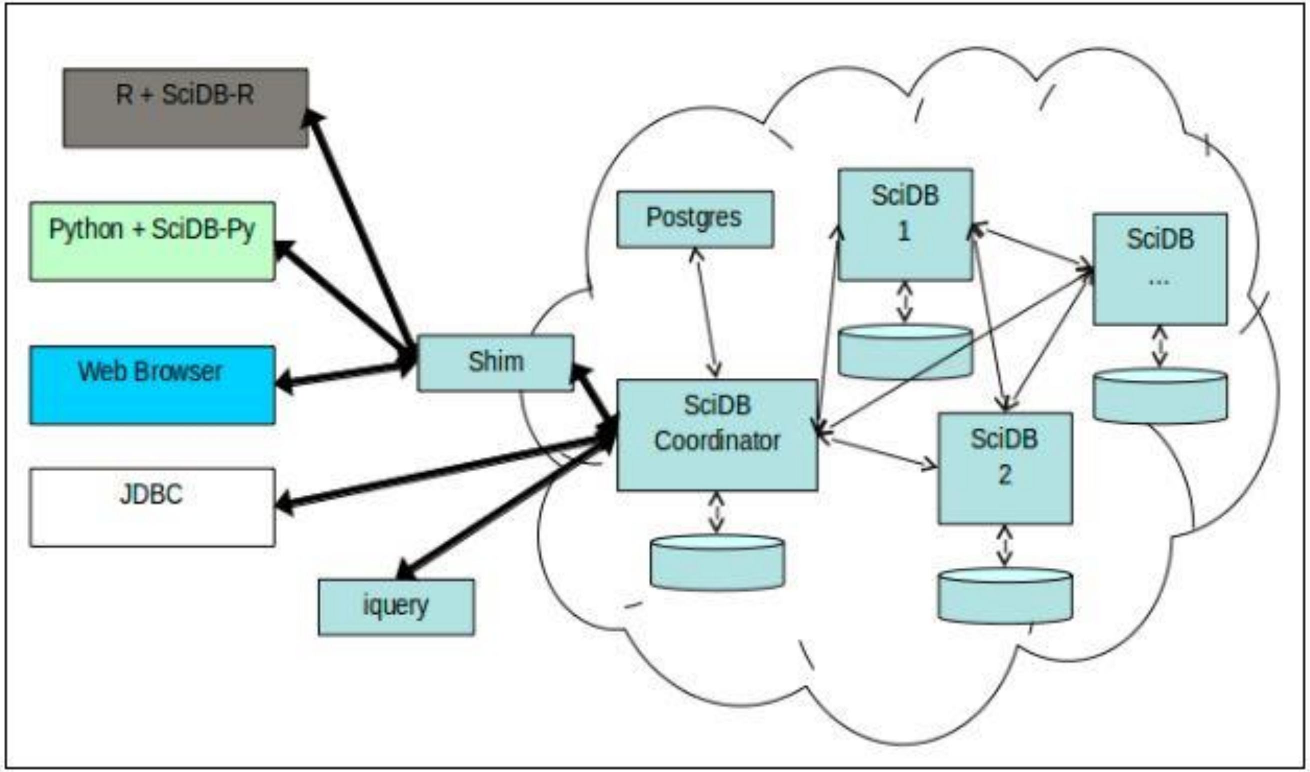
Multidimensional Array Clustering

- Chunks
- User defined co-ordinate system

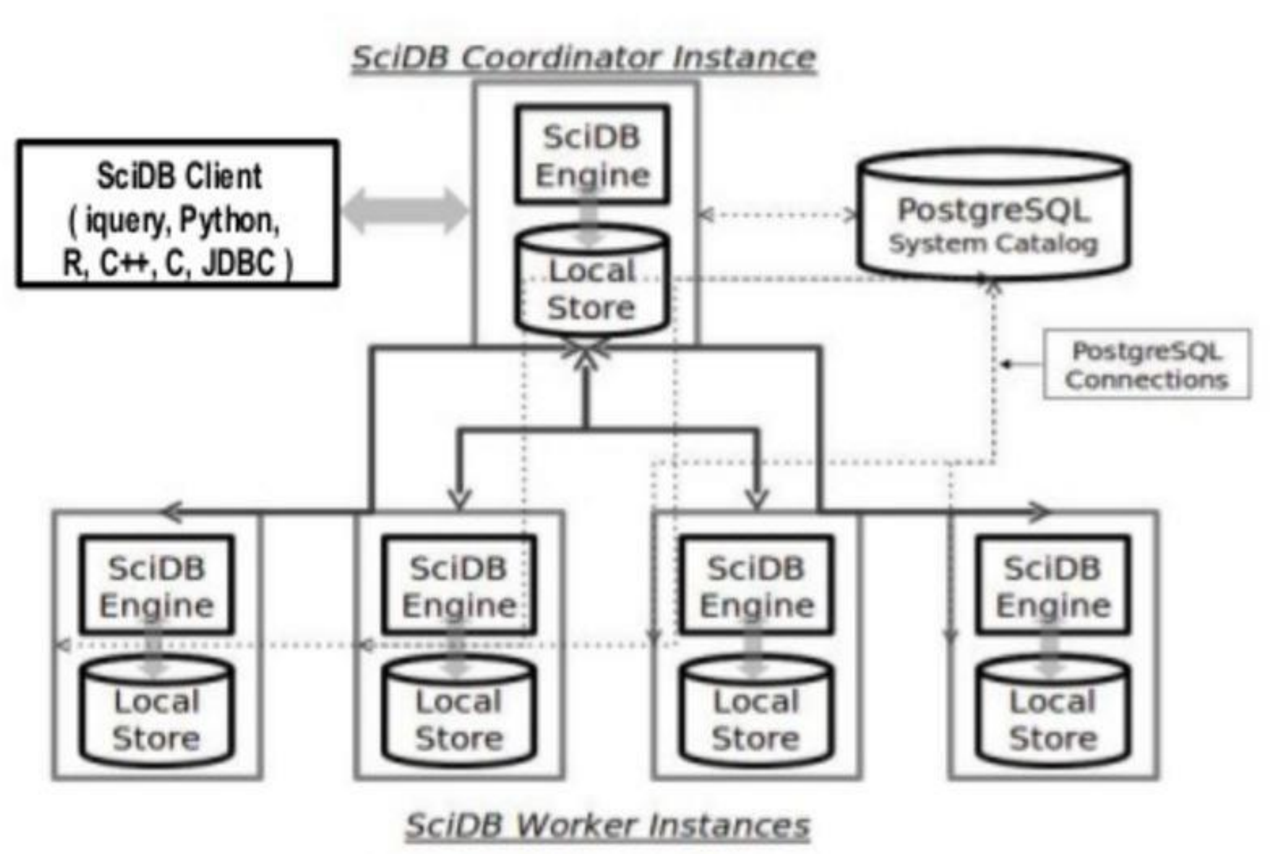
CREATE ARRAY STOCK_MARKET<PRICE: DOUBLE, VOLUME: DOUBLE> [STOCK(string) TIME(datetime)];



Architecture



SciDB System Architecture



Data Type	Default Value	Description
bool	false	Boolean value, true (1) or false (0)
char	\0	Single ASCII character
datetime	1970-01-01 00:00:00	Date and time
datetimetz	1970-01-01 00:00:00 -00:00	Date and time with timezone offset.
double	0	Double-precision floating point number
float	0	Single-precision floating-point number
int8	0	Signed 8-bit integer
int16	0	Signed 16-bit integer
int32	0	Signed 32-bit integer
int64	0	Signed 64-bit integer
string	''	Variable length character string, default is the empty string
uint8	0	Unsigned 8-bit integer
uint16	0	Unsigned 16-bit integer
uint32	0	Unsigned 32-bit integer
uint64	0	Unsigned 64-bit integer

AQL and AFL

- Array Query Language
  - Data Definition Language: create and load arrays
  - Data Manipulation Language: select and operate on data stored in arrays
- Array Functional Language
  - Operators
    - Aggregate
    - Combine
    - Compute
    - Math
    - Rearrange

AQL Examples

- CREATE ARRAY Simple_Array <a1: double,
  a2: int64,
  a3: string>
  [I = 0 : *, 5, 0
  J = 0 : 9, 5, 0];

- SELECT a1 FROM Simple_Array;

- SELECT I FROM Simple_Array;

- INSERT INTO Array1
  Select * from Array2

Color index:
Attributes: a1, a2, a3
Dimensions: I, J
Dimension size: * is unbounded
Chunk size

AFL EXAMPLES

- CREATE ARRAY A <X: double,
  Y: double>
  [I = 0:99, 5, 0];

- CREATE ARRAY B <M: double,
  N: double>
  [I = 0:*, 5, 0 J = 0:99, 5, 0];

- Re-dimensioning array A:
  REDIMENSION_STORE(A, B);

- Aggregate operation: aggregate(A, count(X));

Let's compare

```
CREATE TABLE INPUT_A ( ROW
INTEGER NOT NULL, COL INTEGER
NOT NULL, VAL DOUBLE PRECISION,
PRIMARY KEY ( ROW, COL ) );

CREATE TABLE INPUT_B ( ROW
INTEGER NOT NULL, COL INTEGER
NOT NULL, VAL DOUBLE PRECISION,
PRIMARY KEY ( ROW, COL ) );

CREATE TABLE BASE ( ROW INTEGER
NOT NULL, COL INTEGER NOT NULL,
VAL DOUBLE PRECISION DEFAULT
0.0,PRIMARY KEY ( ROW, COL ) );
```

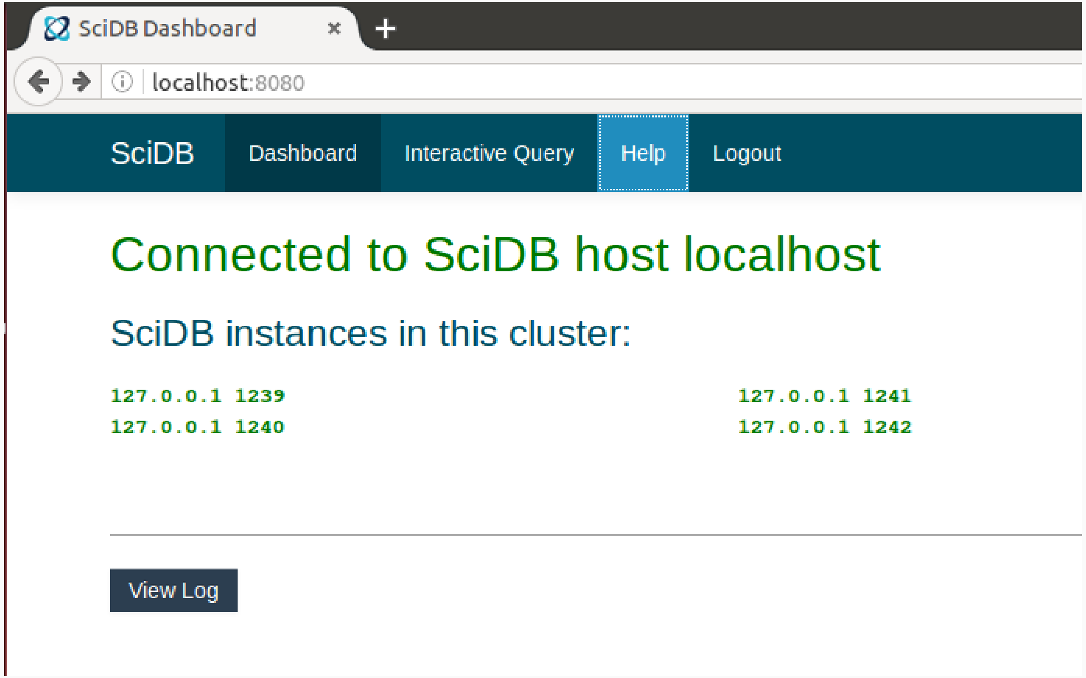
```
WITH MULTIPLY AS ( SELECT A.ROW,
B.COL, SUM ( A.VAL * B.VAL ) AS VAL
FROM INPUT_A AS A
JOIN INPUT_B AS B ON A.COL = B.ROW
GROUP BY A.ROW, B.COL )
SELECT MULTIPLY.VAL + BASE.VAL
FROM MULTIPLY JOIN BASE ON
MULTIPLY.ROW = BASE.ROW AND
MULTIPLY.COL = BASE.COL;
```

Corresponding query in SciDB

```
input_A < val : double >[ row=0:4, col=0:5 ]
input_B < val : double >[ row=0:5, col=0:3 ]
base < val : double>[ row=0:4, col=0:3 ]
gemm ( input_A, input_B, base );
```

SciDB-py

SciDB-py



Python library for SciDB

Easily store and grab arrays

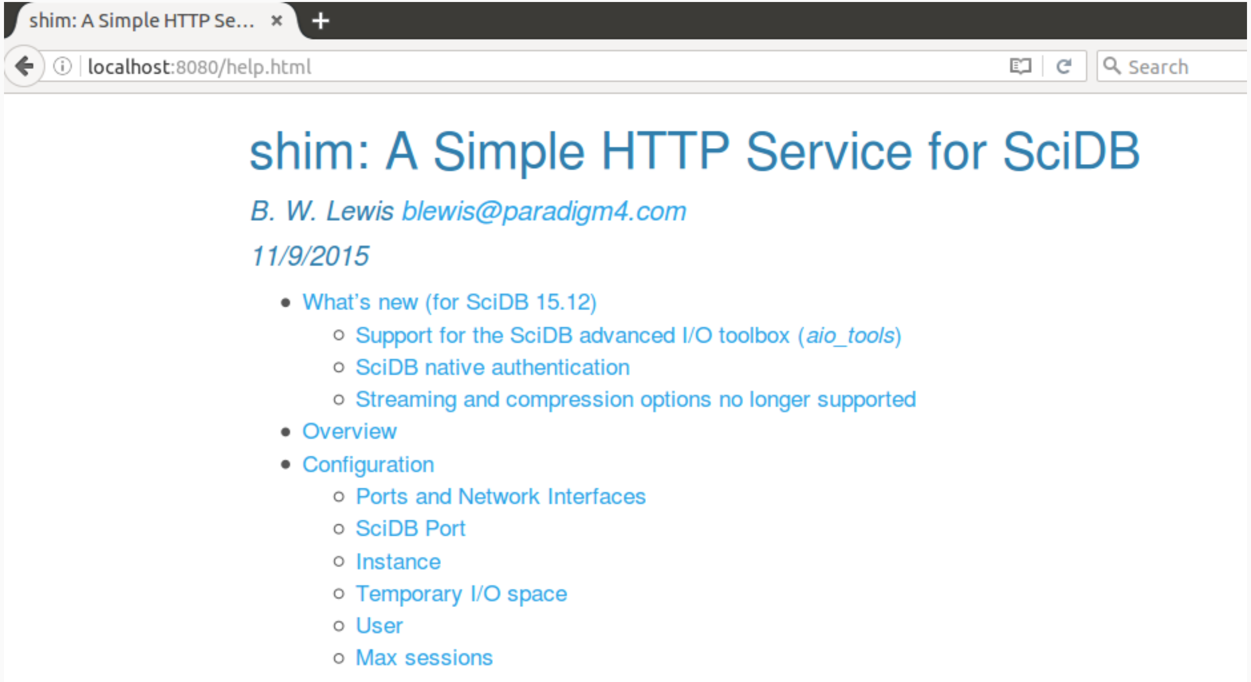
Uses functions to load

SciDB-py Requirements

SciDB installation

Shim (network interface)

Python NumPy



Upload Array to SciDB

- from_array()
- Uploads a numpy array
- Creates a SciDBArray object in python

`random()` to create an array of uniformly distributed random floating-point values:

```
>>> # Create a 10x10 array of numbers between -1 and 2 (inclusive)
>>> #    sampled from a uniform random distribution.
>>> A = sdb.random((10,10), lower=-1, upper=2)
```

`randint()` to create an array of uniformly distributed random integers:

```
>>> # Create a 10x10 array of uniform random integers between 0 and 10
>>> #  (inclusive of 0, non-inclusive of 10)
>>> A = sdb.randint((10,10), lower=0, upper=10)
```

`arange()` to create and array with evenly-spaced values given a step size:

```
>>> # Create a vector of ten integers, counting up from zero
>>> A = sdb.arange(10)
```

`linspace()` to create an array with evenly spaced values between supplied bounds:

```
>>> # Create a vector of 5 equally spaced numbers between 1 and 10,
>>> # including the endpoints:
>>> A = sdb.linspace(1, 10, 5)
```

`identity()` to create a sparse or dense identity matrix:

```
>>> # Create a 10x10 sparse, double-precision-valued identity matrix:
>>> A = sdb.identity(10, dtype='double', sparse=True)
```

Persistent Arrays

- New array functions take an argument called “persistent”.
- Persistent defaults to false.
  - True -> arrays stored in SciDB until removed
  - False -> arrays get removed after python session ended.

Accessing SciDB Array Objects

toarray()

todataframe()

tosparse()

```
[[ 0.19494496  0.01048067  0.29326652  0.55247321]
 [ 0.38221543  0.07841876  0.40206973  0.26819489]
 [ 0.17596939  0.08659856  0.23141057  0.28845458]
 [ 0.4768357   0.5631318   0.88933432  0.96756434]
 [ 0.60846079  0.02796332  0.49568745  0.16120202]]
```

Advantages of Using SciDB-py

- Python
- Aggregates
- No SQL queries
- Much like numpy

SciDB and R

Why R?

- Parallel computing in an easy way.
- Approach naturally fits analytics environment

SciDB package for R

- Two main ways to interact with sciDB
- Use sciDB query language optionally returning results in data frames that can be iterated over.
- Use Array and dataframe like classes in R- statements backed by sciDB arrays

Iquery client

- Iquery executable → basic command line tool for communicating with sciDB

Sample R scripts for genome data

```
library (" scidb ")
library('threejs')
library('ggplot2')

source('/home/scidb/vm_functions.R')
#Will output "creating a generic function for 'image'... that is normal

scidbconnect ()


svded = scidb("KG_VAR_SVD")
# svded is an R representation of SciDB array KG_VAR_SVD

str(svded)
#outputs the structure of the R- representation of the array.
```

Load the sciDB package

Connect to sciDB

Sample R scripts

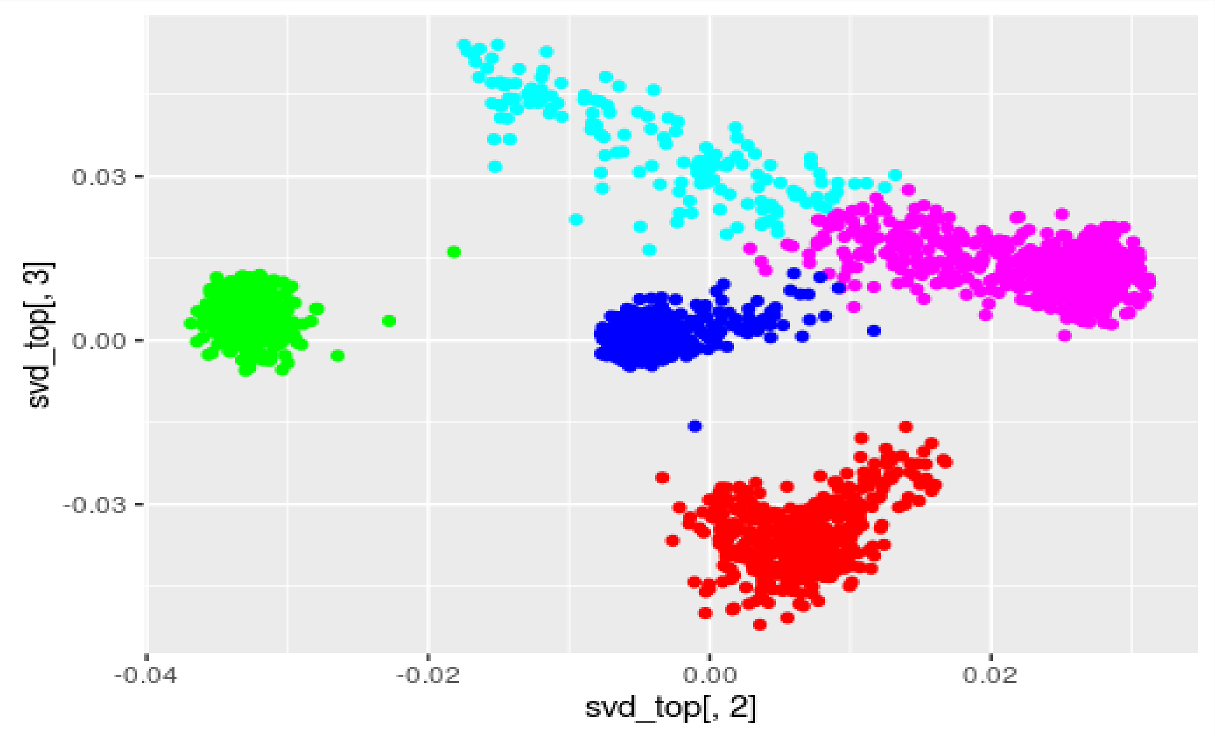
```
#Download just the 3 left vectors into R and make a matrix out of them:
svd_top = df2xyvm(iqdf(subset(svded, i<=2), n=Inf))

#Do kmeans clustering of these vectors in R now:
clustering = kmeans(svd_top, 5, nstart=50)

#Convert the kmeans cluster assignments to colors
color=gsub("[0-9]","",palette()[clustering$cluster+1])

#The relative distance between the dots is a measure of "genetic
closeness"
print(qplot(x=svd_top[,1], y=svd_top[,2], color=I(color)))
#Vectors 1 and 3
qplot(x=svd_top[,1], y=svd_top[,3], color=I(color))
#Vectors 2 and 3
qplot(x=svd_top[,2], y=svd_top[,3], color=I(color))
```

Sample R scripts



Advantages SciDB-R

- Use SciDB as back-end database
- Use SciDB to offload large computations to cluster.
- Use SciDB to filter and join data before performing analytics
- Use SciDB to share data among multiple users.
- Use SciDB to perform multi-dimensional windowing and aggregation.

Popular Applications

Early use cases - Resulted in birth and initial steps of SciDB

- Satellite Imagery
- Astronomy
- Genomics

Satellite Imagery - MODIS data

- Raw imagery of Earth data is a 3D array.
- Need to be fed into high level applications.
- Usually, the result is not satisfactory.

Astronomy - LSST data

- Telescope records images as 2D array.
- Lyra astronomy project needs a common repository for multiple telescopes.
- Need to be fed into high level applications.

Genomics

- Complete genome for a single human - 2D array
- Will be compared against human disease characteristics
- Biclustering in a large data set implemented in R vs implemented in SciDB-R showed significant differences.

Popular Applications

More refined uses - Resulted in growth of SciDB

- 1000 Genomes Browser
- LUX detector data
- Brazilian rainforests' research

1000 Genomes Browser by NCBI

- Theoretically, genotype data can be a 2D array.
- Output of querying this data set is typically all columns for a row, or all rows for a column obtained by using slice and between operations.
- Thus, array form of SciDB enables complex combinations of filter and cross_join queries.

LUX Detector by NERSC

- To gather evidence about the interaction between dark matter and normal matter.
- Represented as a 3D array, with 50 data attributes per cell.
- Complex queries involved like regrid, filter and cross_join.
- Using SciDB, entire analysis on 600,000,000 pulses took 4 hours.

Brazilian Rainforests' Research by INPE

- An attempt to reproduce a controversial finding published by a different team.
- MODIS HDF-5 data set containing visible and infrared bands covering Brazil was used.
- Represented as 3D array - 7 TB data.
- SciDB took 4.6 hours to reproduce the finding.

Paradigm 4 customers:

FOUNDATION MEDICINE

Complete GENOMICS A BGI Company

HARVARD MEDICAL SCHOOL

DEPARTMENT OF BIOMEDICAL INFORMATICS

janssen

GENOMICS PLC

agios

BERKELEY LAB

Lawrence Berkeley National Laboratory

DARPA

LINCOLN LABORATORY

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

NASA

NIH

Advantages of SciDB

- Keeps all the data
- Fast computation time
- Multiple instances
- No set data format
- Returns window query results in constant time

Advantages of SciDB over other systems

- RDBMS: Array system instead of tables.
  - Fast data regridding
  - In-situ linear algebra operations
  - Science-appropriate operators in AQL
  - Support for ‘never discard data’ policy of the scientific data users
  - Can store uncertain nature of the scientific data
  - Multiple types of “null” operator

Advantages of SciDB over other systems(contd):

- File System:
  - Metadata is not needed to be stored separately
  - Usual DBMS operations are used.
  - Exact layout of the file system is not needed to be known.
- Hadoop:
  - Has an efficient communication model
  - Not vulnerable to scalability issues

Disadvantages of SciDB

- Keeps all the data
- Small community
- Can’t organize arrays and metadata
- Not useful in small industries, small datasets and structured data
- Sparse dataset

Verdict for SciDB

To Use or Not To Use?

References

1. https://paradigm4.atlassian.net/wiki/display/ESD/SciDB+Reference+Guide
2. http://www.paradigm4.com/technology/multidimensional-array-clustering/
3. http://www.paradigm4.com/HTMLmanual/14.12/scidb_ug/
4. https://arxiv.org/ftp/arxiv/papers/1103/1103.3863.pdf
5. http://ieeexplore.ieee.org/document/6461866/
6. http://scidb-py.readthedocs.io/en/stable/
7. http://discover.paradigm4.com/scidb-database-for-21st-century.html

Thank you


HAVE ANY QUESTIONS
DO YOU?