DATAJAGUAR

JaguarDB

New NoSQL Database

Jonathan Yue, Ph.D.
Founder of DataJaguar

Presentation at NoSQL Meetup
Feb 28, 2017, Penisula, Silicon Valley

DATAJAGUAR

Overview

Data is distributed

- All Masters
- No Slaves

Overview

SQL-like query statements

```
jaguar>  create table comp  ( key: city char(8), cn char(8),  value: emp int, ct char(2) );
jaguar>  insert into comp values ( ‘SF’, ‘CSCO’, 10000, ‘HP’ );
Jaguar> select city, cn from comp where city=‘SF’;
Jaguar> select city, count(1) as cnt from comp group by city order by cnt;
Jaguar> update comp set emp=20000 where city=‘SF’ and cn=‘CSCO’;
Jaguar> delete from comp where city=‘SF’ and cn=‘test111’;
```

Overview

Programming Languages

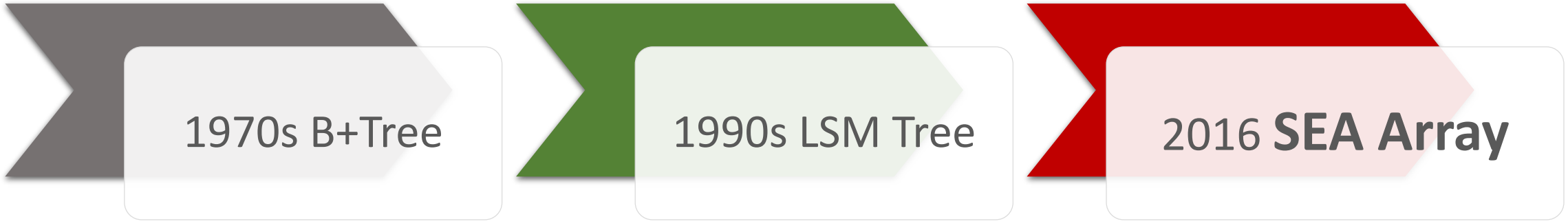
Jaguar Server: C++

Jaguar Client: Java, Python, C++, Php, Scala, Ruby, NodeJS, Go
JDBC, Spark

DATAJAGUAR

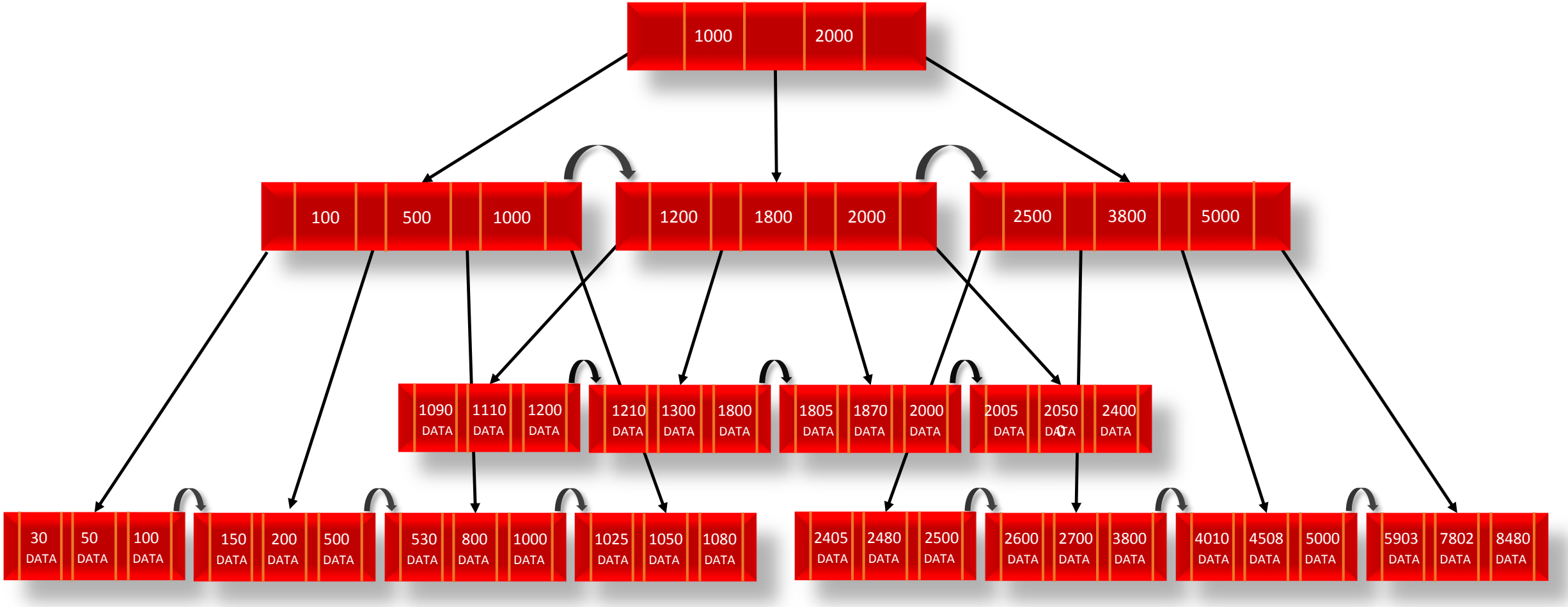
Why JaguarDB

JaguarDB uses a new data structure



DATAJAGUAR

B+Tree Index



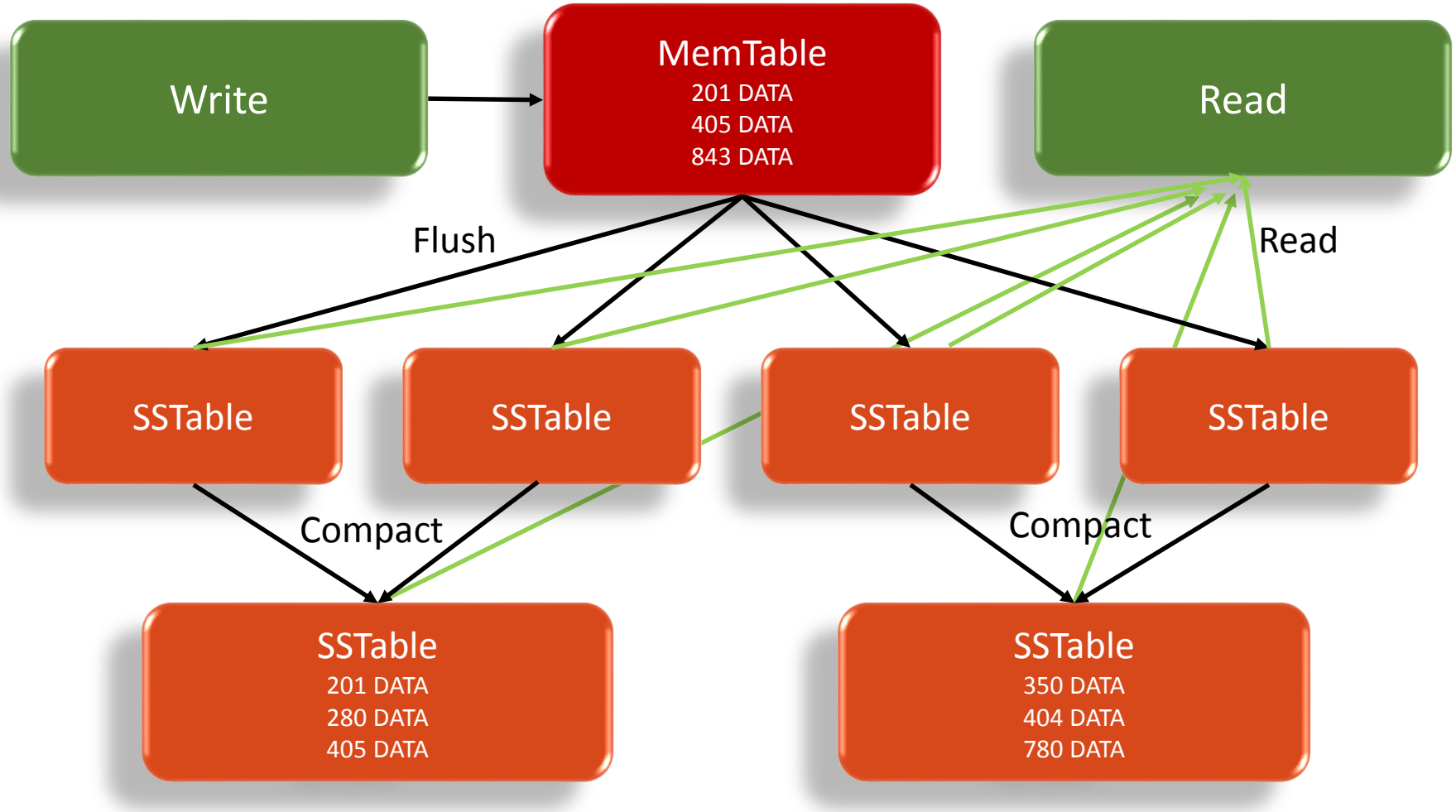
B+Tree 1970

MySQL, Oracle, PostgreSQL, MongoDB

- Many node splits
- Leaf nodes are linked list
- Index scan requires disk seeks

DATAJAGUAR

LSM Tree



LSM Tree 1990

Cassandra, HBase, BigTable

- MemTable and SSTable
- Table compaction
- Requires partition key

What Is SEA

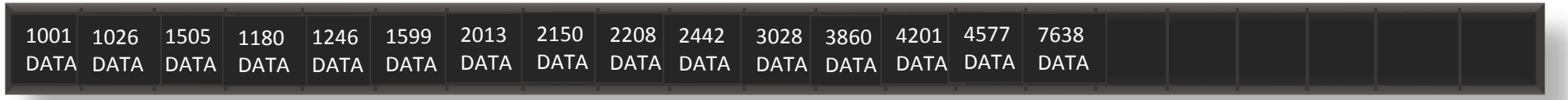
Sorted Elastic Array (SEA)

- Data sorted in simple flat array with open spaces
- Sparse array (not so sparse) ~30% sparse

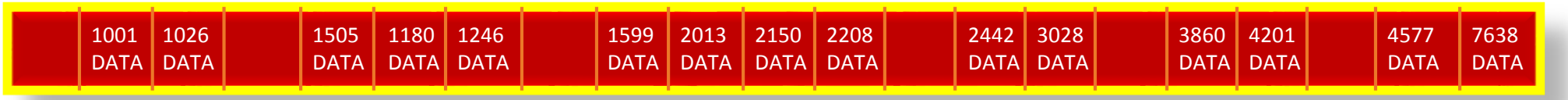
Regular Array Vs. SEA

Array is good for data locality, read-ahead, caching

Conventional Sorted Array Insert $O(N^2)$

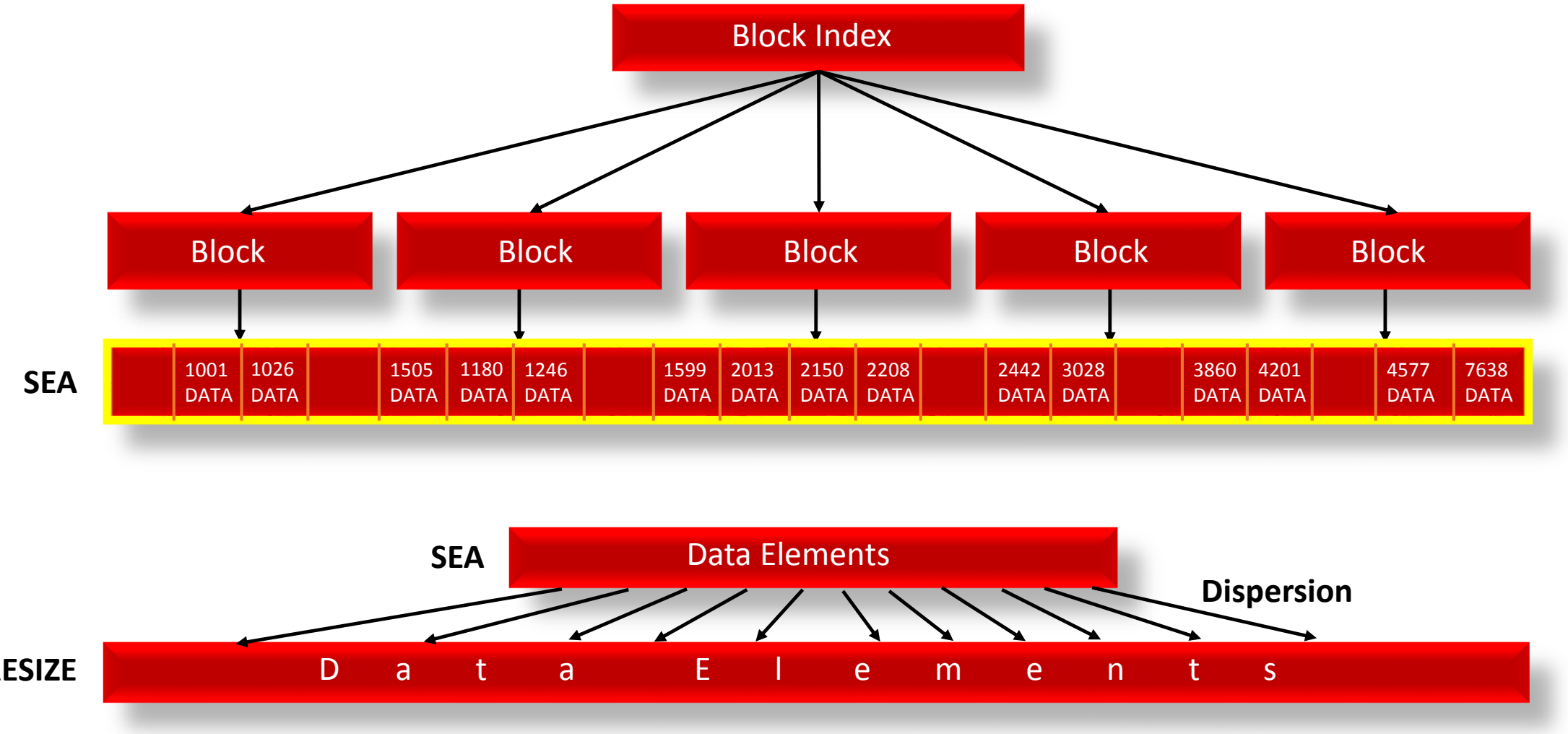


Sorted Elastic Array (SEA) Insert O(NlogN)



DATAJAGUAR

Sorted Elastic Array (SEA) in JaguarDB



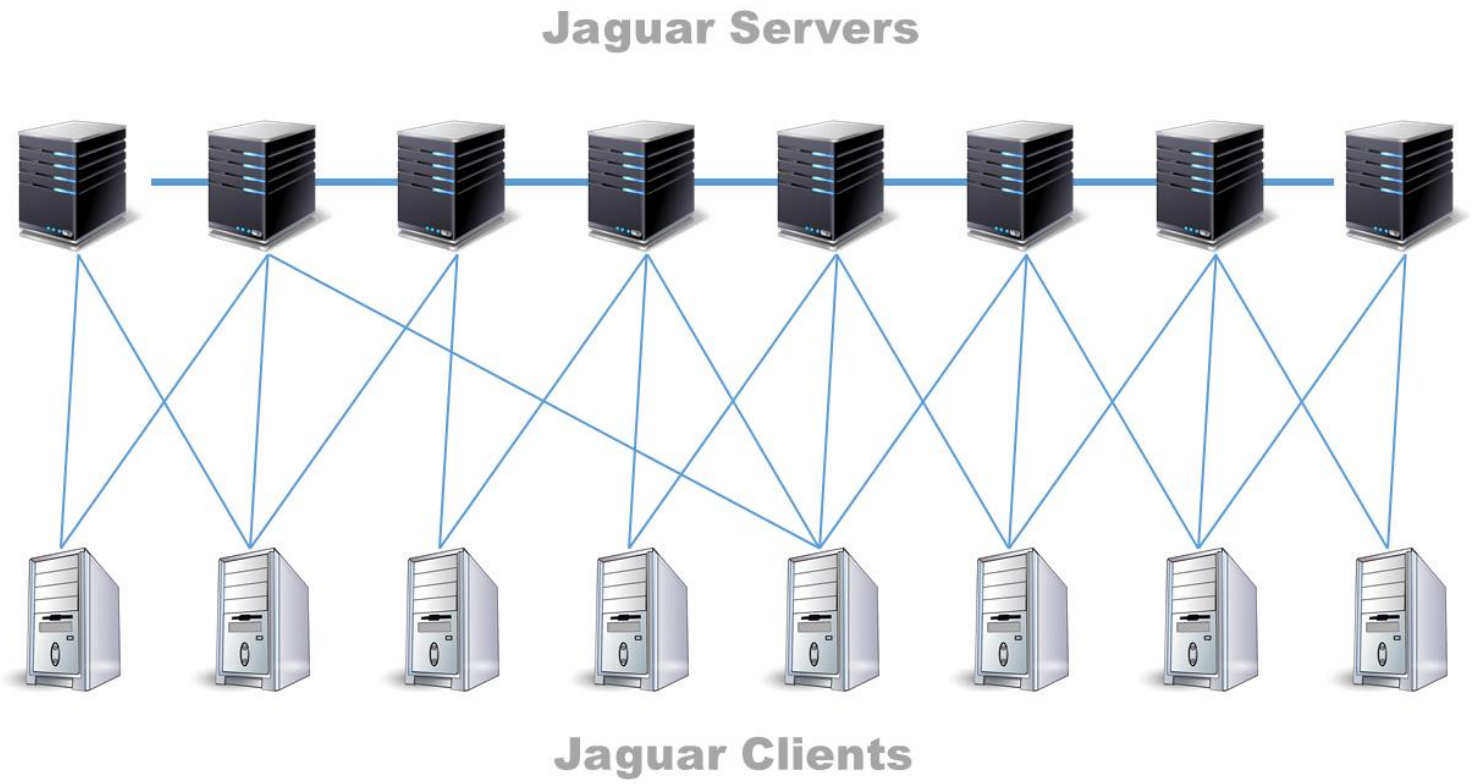
DATAJAGUAR

Properties of SEA

- Fast sequential index scan
- Less data fragmentation
- Good data locality
- Read-ahead efficiency
- Works well for both HDD and SSD

DATAJAGUAR

Architecture



- Server scale out
- Client scale out

DATAJAGUAR

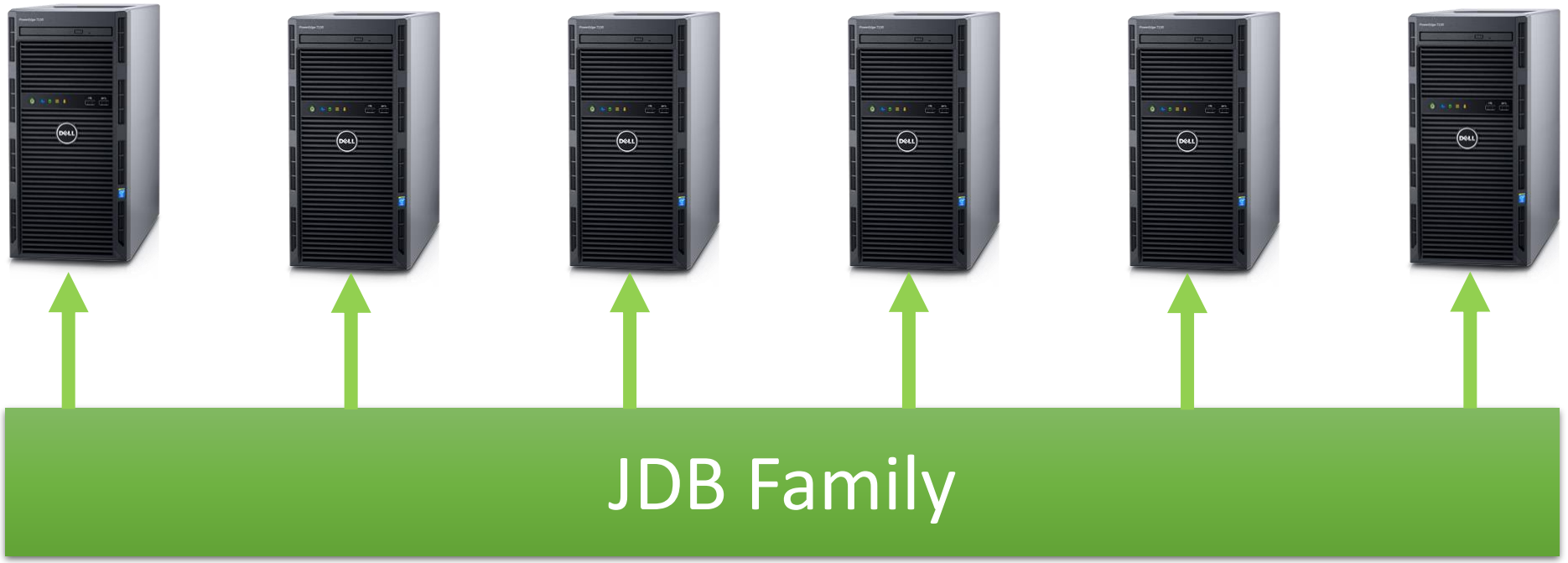
Architecture

Choosing a server to save data

DATAJAGUAR

File Structure

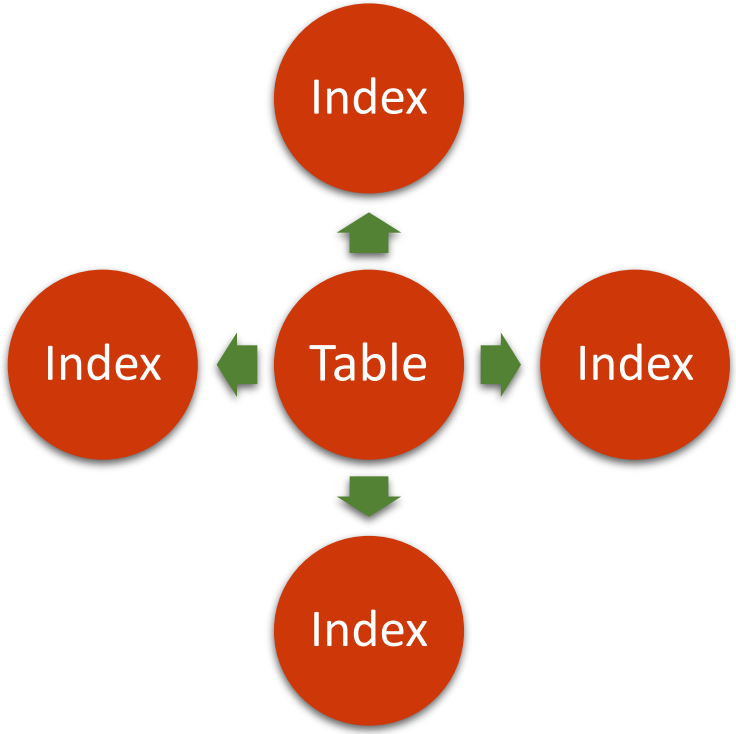
Data records are saved in a family of JDB files



DATAJAGUAR

Seconday Indexes

One Table -- Multiple Indexes

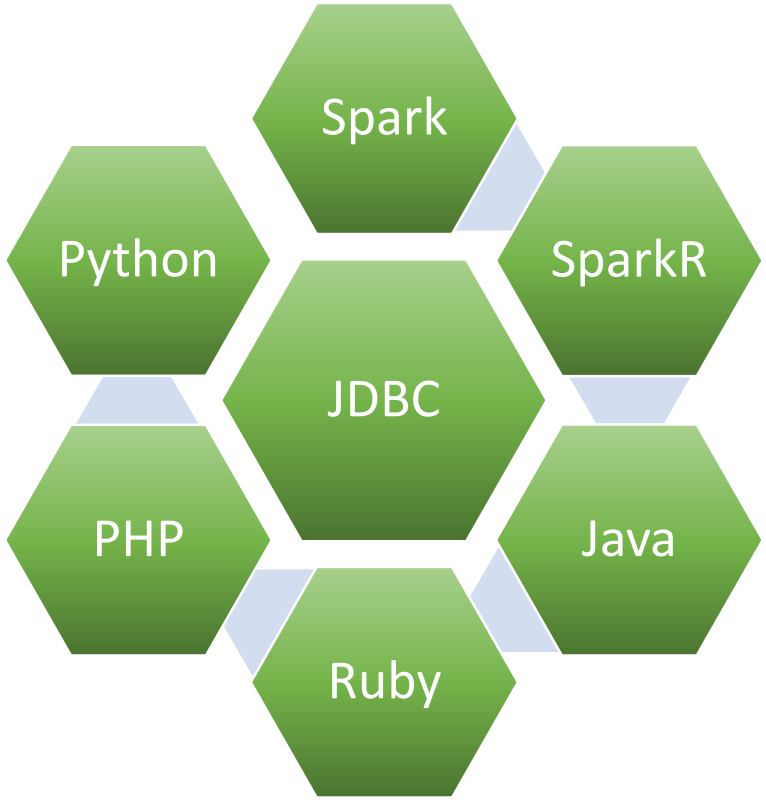


- Each attribute can have a index
- Significant attributes in index
- Direct search on indexes

DATAJAGUAR

Big Data Ecosystem

Integration with other components



Easy Installation

Two steps on each host

1. `Execute install.sh`

2. `Edit conf/host.conf`

SQL Like API

Supports shell, API of Java, C++, Python, PHP, …

```
create table member (key: uid char(32), value: phone
bigint, addr char(128), zip int );

insert into member values ('jdoe8888', 4088882228, '123
Main Street, SF, CA92333', 92333 );

select * from member where uid='jdoe8888';
```

JagguarDB supports 90+ SQL-like commands and functions

Peformance

Write

- 100 million records
- 1000 bytes in each record
- 3 servers
- ➔Used 100 Minutes
- 1.8T HDD, 72GB RAM, 16 Core, 1Gbps, Linux

Work In Progress

- Fault Tolerance (one or two data replicates)
- Scaling (adding new nodes need no data migration)

Application

IoT data, machine data, sensor data, process data

-- speed

-- location

-- IP address

-- temperature

-- acceleration

-- blood pressure

-- heart beat

-- stock price

-- moisture …

Thank You

www.datajaguar.com

Jonathan Yue

www.linkedin.com/in/jonyue