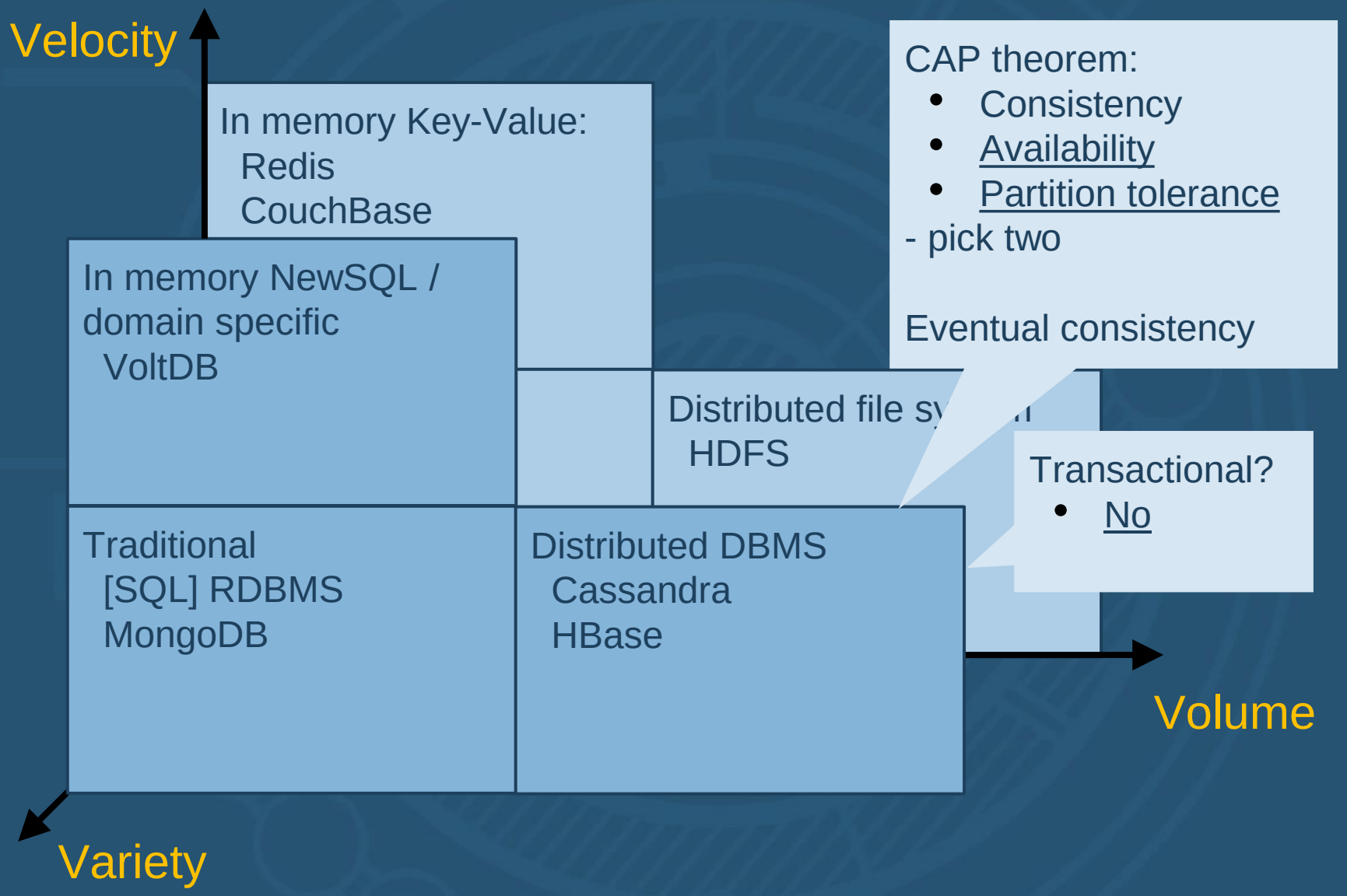
Sperasoft

Your game dev partner

Developing with Cassandra

Choose the DB



Why select Cassandra?

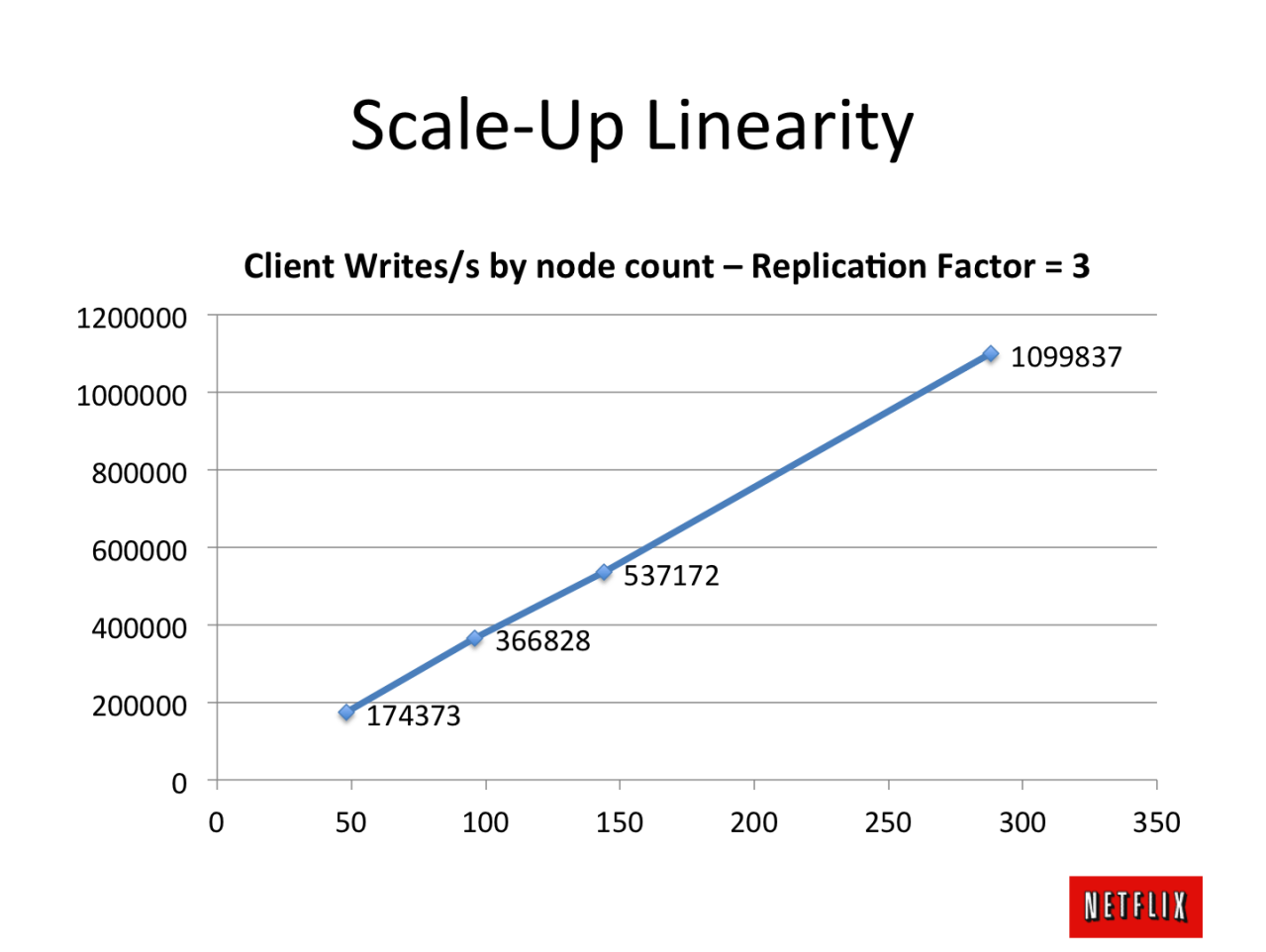
- [relatively] easy to setup
- [relatively] easy to use
- ~zero routine ops
- it works (!!) as promised:
    - real-time replication
    - node/site failure recovery
    - zero load writes
    - double of nodes = double of speed

Because Cassandra is Fast!

But needs some time to deliver

- *12'000 WPS on a laptop*
- *~0.1 / 1 ms constant latency for writes/reads*

Scalability



Check References:

http://techblog.netflix.com/2011/11/benchmarking-cassandra-scalability-on.html

http://vldb.org/pvldb/vol5/p1724_tilmannrabl_vldb2012.pdf

Good and Not So Good

Good For:

- **log-like data**
  TTL helps
- **massive writes**
  1M WPS enough?
- **simple real-time analytics**

Not So Good For:

- **dump of junk**
  (consider HDFS)
- **OLAP**
  (depends on "O")

Apache Cassandra

Sperasoft
Your game dev partner

Distributed DBMS

Just DBMS - closed monolithic solution

- not a platform to run custom code (as MongoDB);
- not an extension (as HBase);
- highly optimized

No-master, eventually consistent

NoSQL

Data model - Key-Value

http://cassandra.apache.org

History

Developed at Facebook for Inbox search

Released to open source in 2008

In use:

- **Netflix** - main non-content data store~500 Cassandra nodes *(2012)*
- **eBay** - recommendation system"dozens of nodes", 200 TB storage *(2012)*
- **Twitter** - tweet analysis100 + TB of data
- **More clients:** (http://www.datastax.com/cassandrausers)

Mature & Agile

1.0 - October 2011

1.1 - April 2012

1.2 - January 2013

2.0 - expected this summer (2013)

June 26 2013 - 158 bugs, 89 worth to notice

Sperasoft Experience:

- hit 1 bug in production (stability issue)
- hit 1 bug in QA (in a crafted case)

Distributions

Apache .tar.gz and Debian

packageshttp://cassandra.apache.org/download/

DataStax DSC - Cassandra + OpsCenter

http://planetcassandra.org/Download/DataStaxCommunityEdition

Embedded – for funct. tests on Java apps

Maven

Documentation

http://wiki.apache.org/cassandra/

http://www.datastax.com/docs

What Hardware?

CPU: ARM 700 MHz
RAM: 500 MB
Storage: SD card
Price: $25

200 WPS!

http://www.slideshare.net/planetcassandra/5-andy-cobley-raspberry-pi

More on Hardware

Bare metal

CPU: 8 cores (4 works too)

RAM: 16 - 64 GB (min 8 GB)

Storage: rotating disks 3 - 5 TB total (SSD better)

VM works too, but...

Storage: local disks, avoid NAS

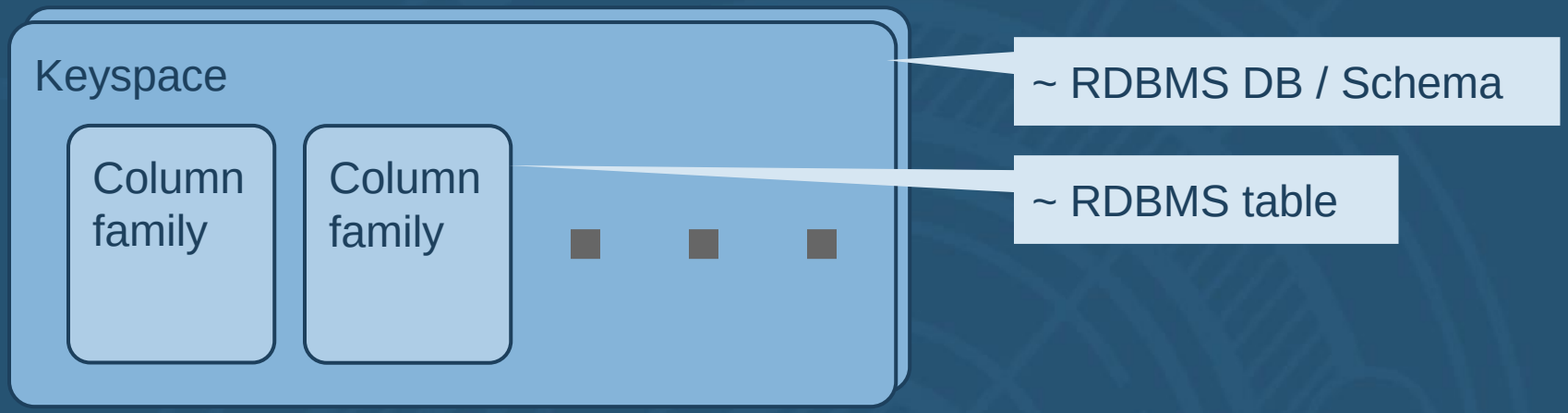
Software Yes & No (production)

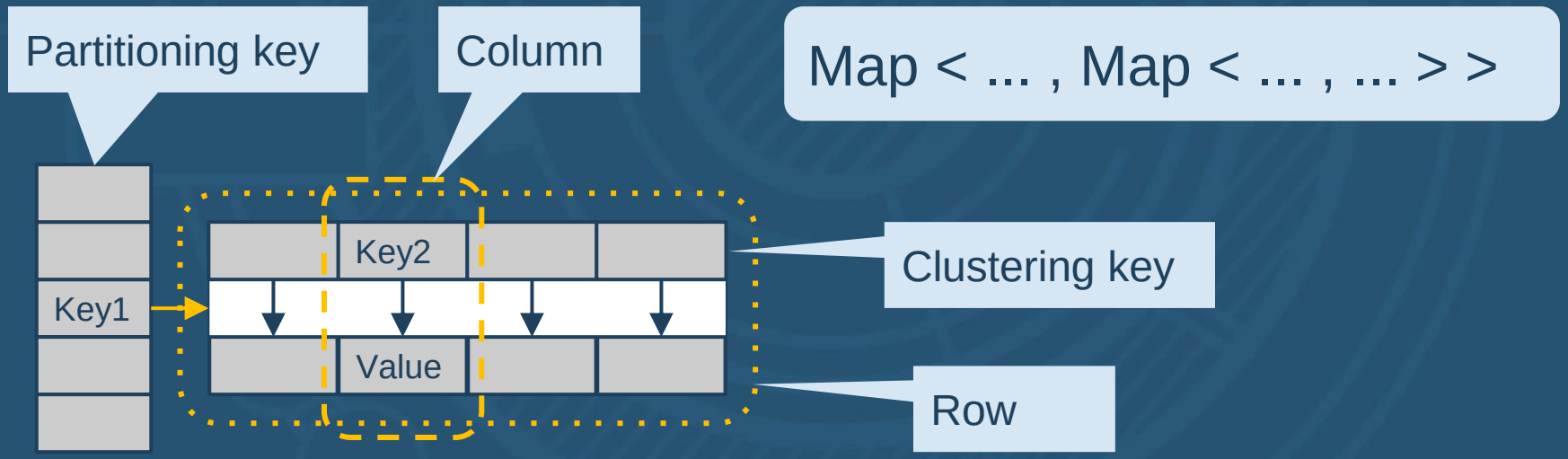
Software for Development: All Yes

OpenJDK

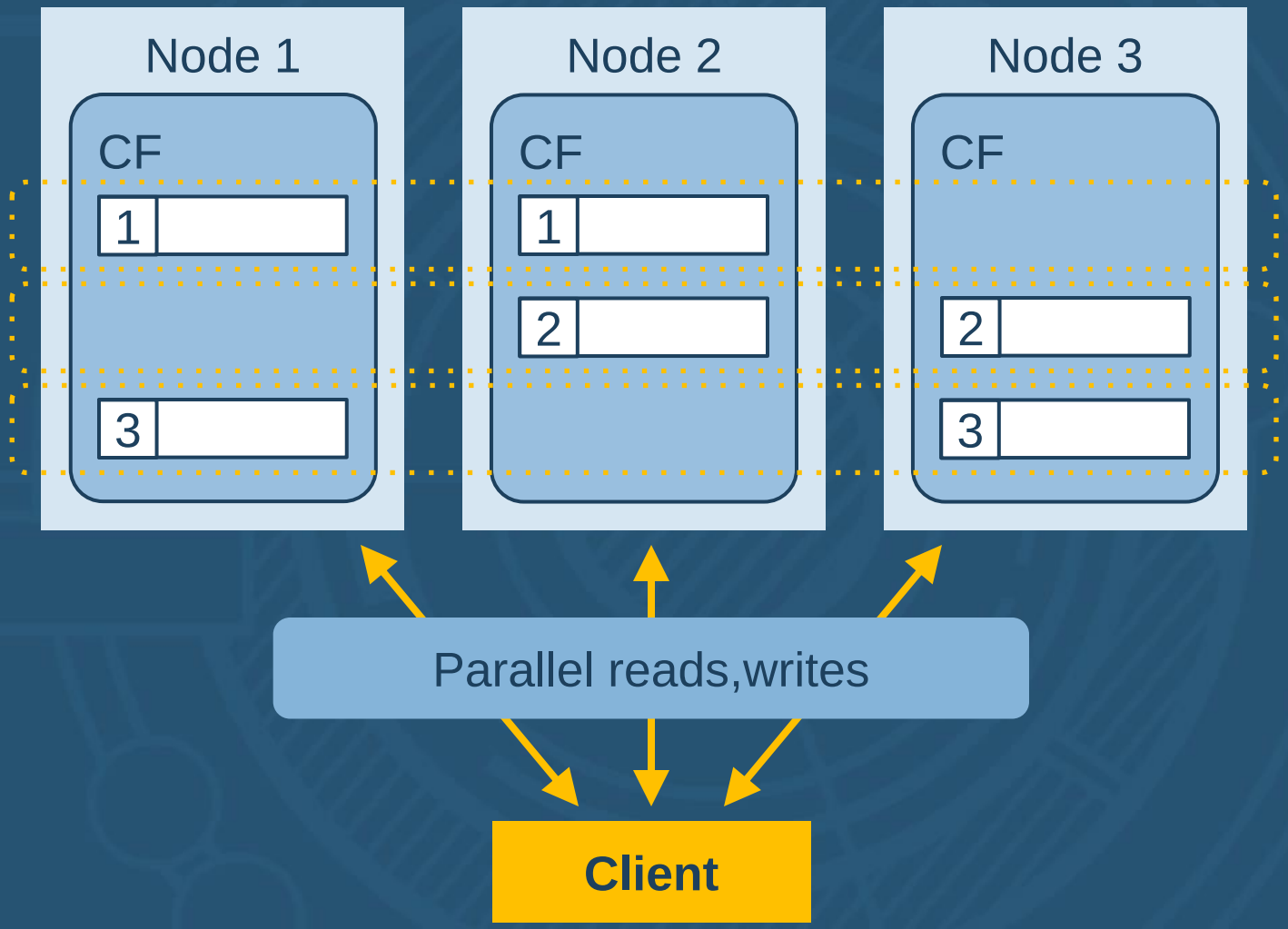
Plus more: *Java, Python, C#, PHP, Ruby, Clojure, Go, R, ...*

Data Model


Keyspace
Column family
Column family
~ RDBMS DB / Schema
~ RDBMS table


Partitioning key
Column
Map < ... , Map < ... , ... > >
Key1
Key2
Value
Clustering key
Row

Data on Discs



Client API Options

Thrift RPC	Native protocol + CQL3
Apache Thrift	Custom protocol
Synchronous	Asynchronous
Schema-less	Static schema
Store & Forward	Cursors promised in 2.0
API for any language	Java; Python, C# coming
Cryptic API	JDBC-like API
Supported yet	Going forward

https://github.com/datastax/java-driver

Data Modeling for NoSQL

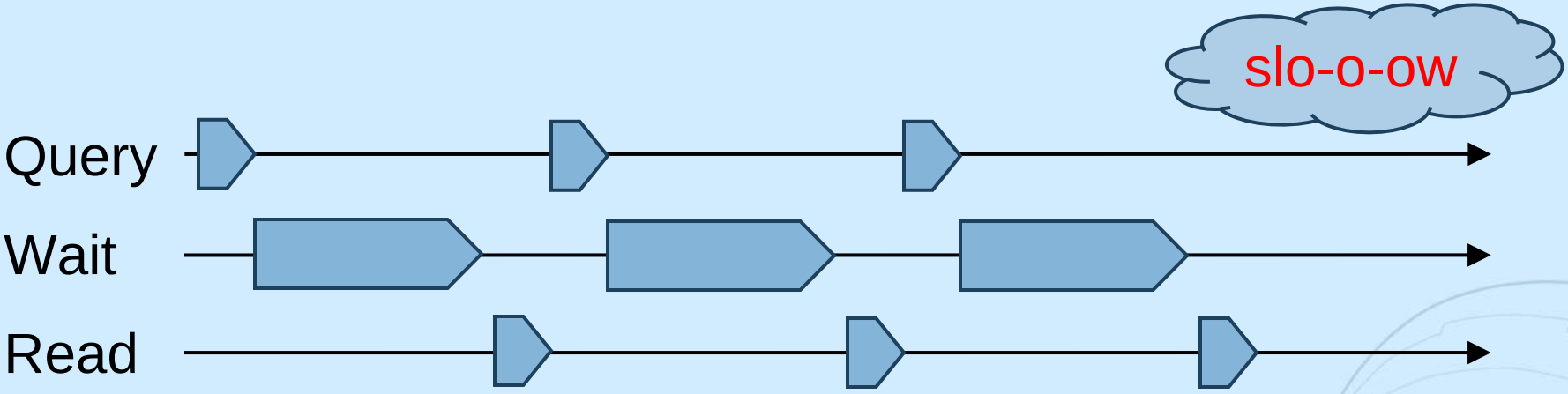
- **Forget RDB design principles**
- **Forget abstract data model**
  - shape data for queries
- **No joins - materialized views**
- **Data duplication - OK**
- **Remember eventual consistency**
- **Queries are precious**
- **Use right data types - timestamp, uuid**

Why? Because NoSQL is a low level tool for high optimization.

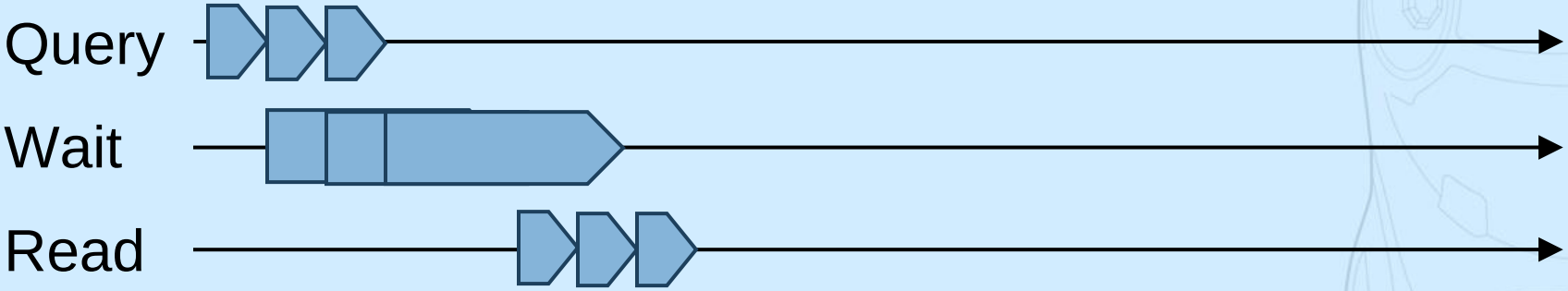
Do & Don't

Patterns and Anti-patterns

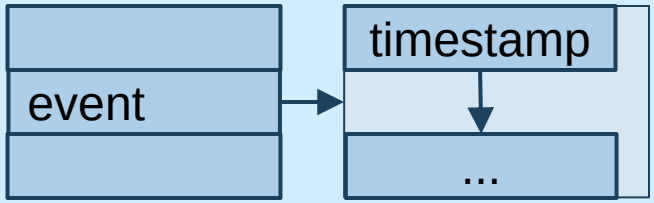
Sequential Read



Parallel it:



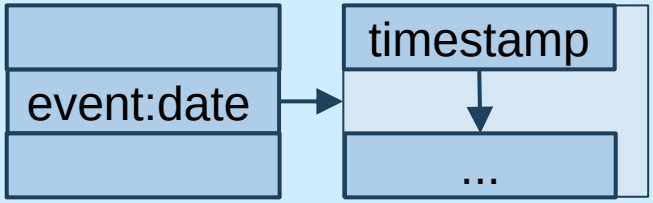
Timeline




```
CREATE TABLE timeline (
        event           uuid,
        timestamp timeuuid,
        ...
        PRIMARY KEY (event, timestamp)
);
```

slo-o-ow

Long rows - Cassandra handle 2G columns, but...




```
CREATE TABLE timeline (
        event           uuid,
        date                    long,
        timestamp timeuuid,
        ...
        PRIMARY KEY ((event, date), timestamp)
);
```

Still bad - need sharding

http://www.datastax.com/dev/blog/advanced-time-series-with-cassandra

Plan Data Immutable

Insert = Update = Delete

	a	b	c	d
1	A	B	C	D
1		Y		
1			Z	
1				✖
1	A	Y	Z	

UPDATE ... SET b = 'Y' WHERE id = 1

INSERT INTO ... SET (id, c) values (1, 'Z')

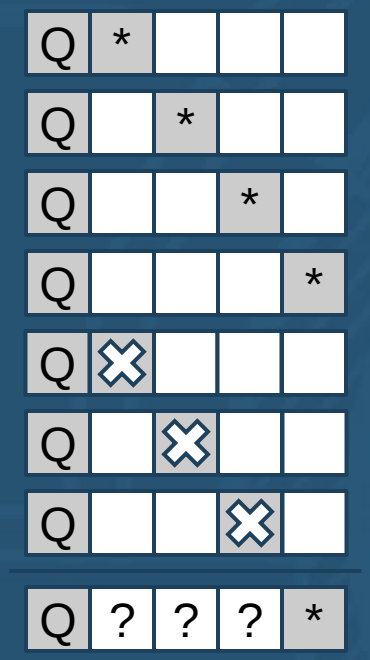
DELETE d FROM ... WHERE id = 1

SELECT * FROM ... WHERE id = 1

have to fetch 4 rows

slo-o-ow

Queue



Queue: INSERT INTO ...
SET(name, enqueued_at, payload)
VALUES ('Q', now(), ...)

Dequeue: DELETE payload FROM ...
WHERE name = 'Q'
AND enqueued_at = ...

Pick the next: SELECT * FROM ...
WHERE id = 1 LIMIT 1

have to fetch 4 rows

slo-o-ow

http://www.datastax.com/dev/blog/cassandra-anti-patterns-queues-and-queue-like-datasets

How Many?

```
SELECT count(*)
        FROM ... WHERE .... ;
```

Full scan over the selec

=>

Default 10'000 rows limit

=> wrong count

slo-o-ow

mess

Have an integer column and increment it

Remember - eventual consistency.

Concurrent updates

=> wrong count

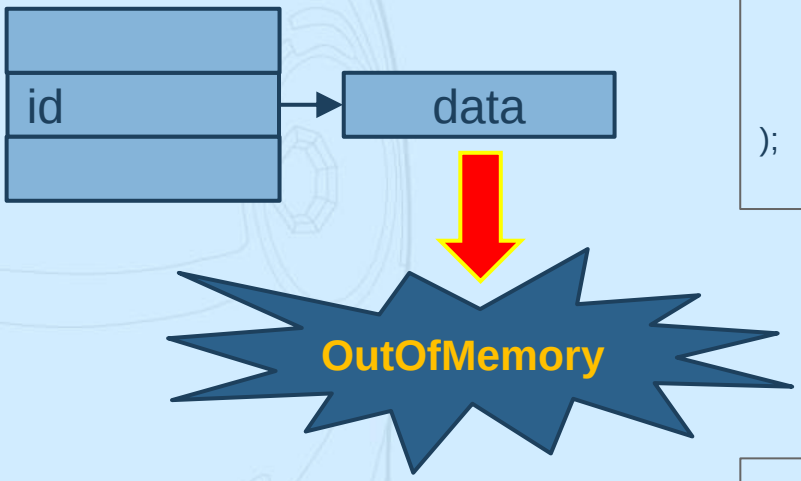
mess

```
CREATE TABLE count_table (
        id
        uuid,
        value                    counter,
        PRIMARY KEY (id)
);
...
UPDATE count_table
    SET value = value + 1
        WHERE id = ... ;
```

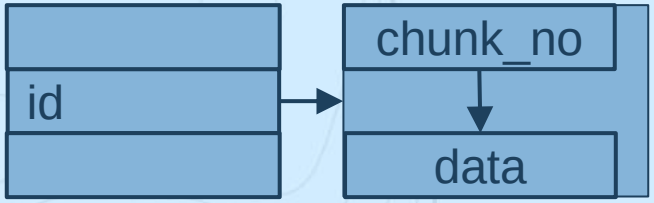
Counter column family

http://www.datastax.com/documentation/cassandra/1.2/cassandra/cql_using/use_counter_t.html

Blobs



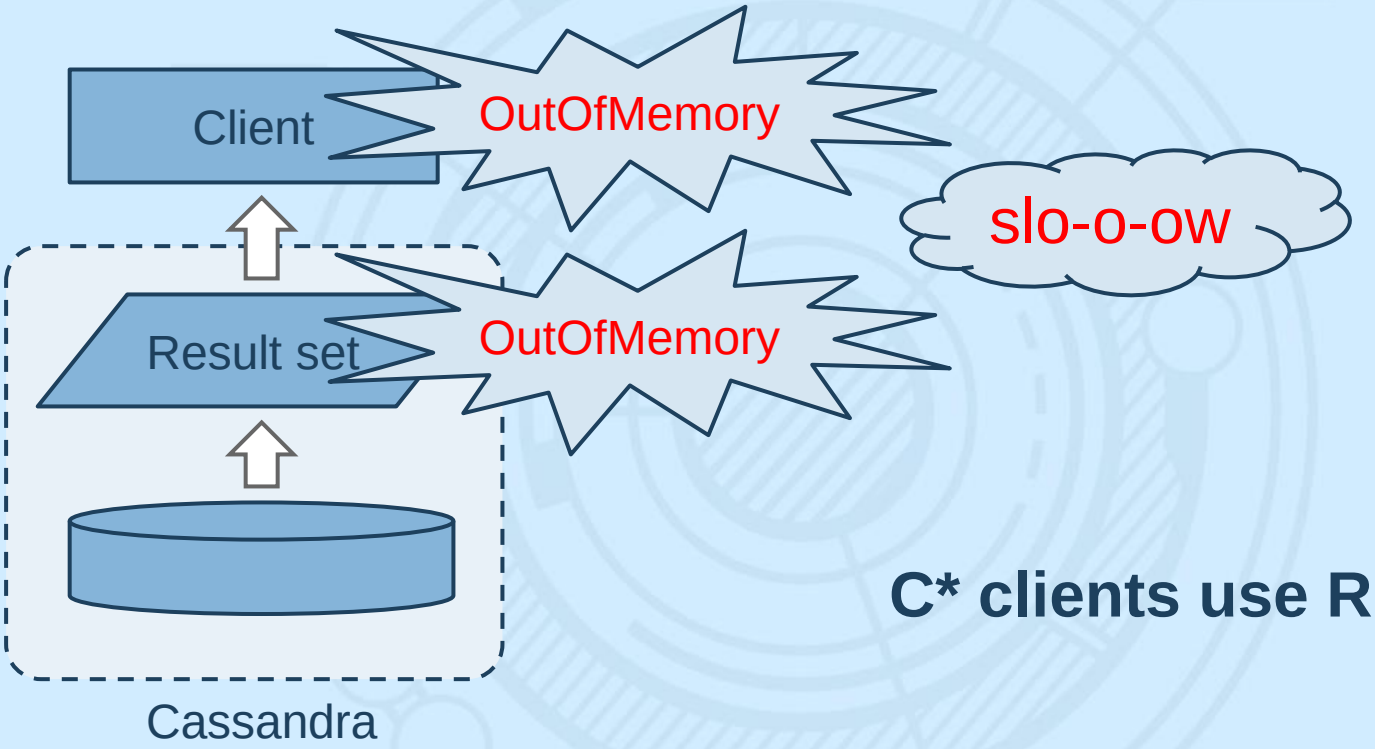

```
CREATE TABLE blob (
        id                      uuid,
        data                    blob,
        PRIMARY KEY (id)
);
```




```
CREATE TABLE blob (
        id                      uuid,
        chunk_no    int,
        data                    blob,
        PRIMARY KEY (id, chunk_no)
);
```

http://wiki.apache.org/cassandra/FAQ#large_file_and_blob_storage
http://wiki.apache.org/cassandra/CassandraLimitations

Unbounded Queries



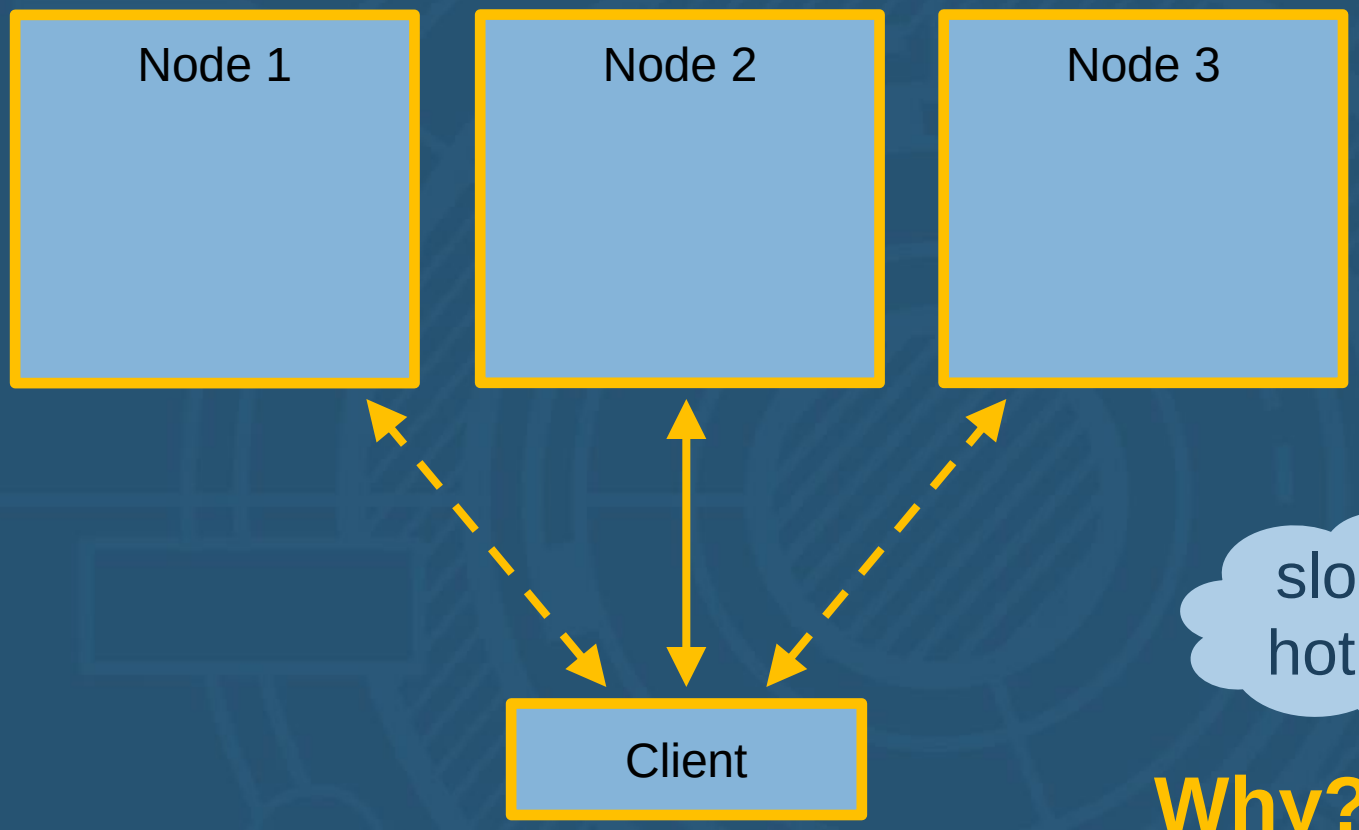
C* clients use RPC yet

Cursors. Planned to 2.0.

https://issues.apache.org/jira/browse/CASSANDRA-4415

Column names - data... yet Keep them short.

Limit Client to a node



slo-o-ow
hot spots

Why??

Helpful Links

Sperasoft @ slideshare: *http://www.slideshare.net/Sperasoft*

Sperasoft @ speakerdeck: *https://speakerdeck.com/sperasoft*

Sperasoft @ github: *https://github.com/Sperasoft/Workshop*