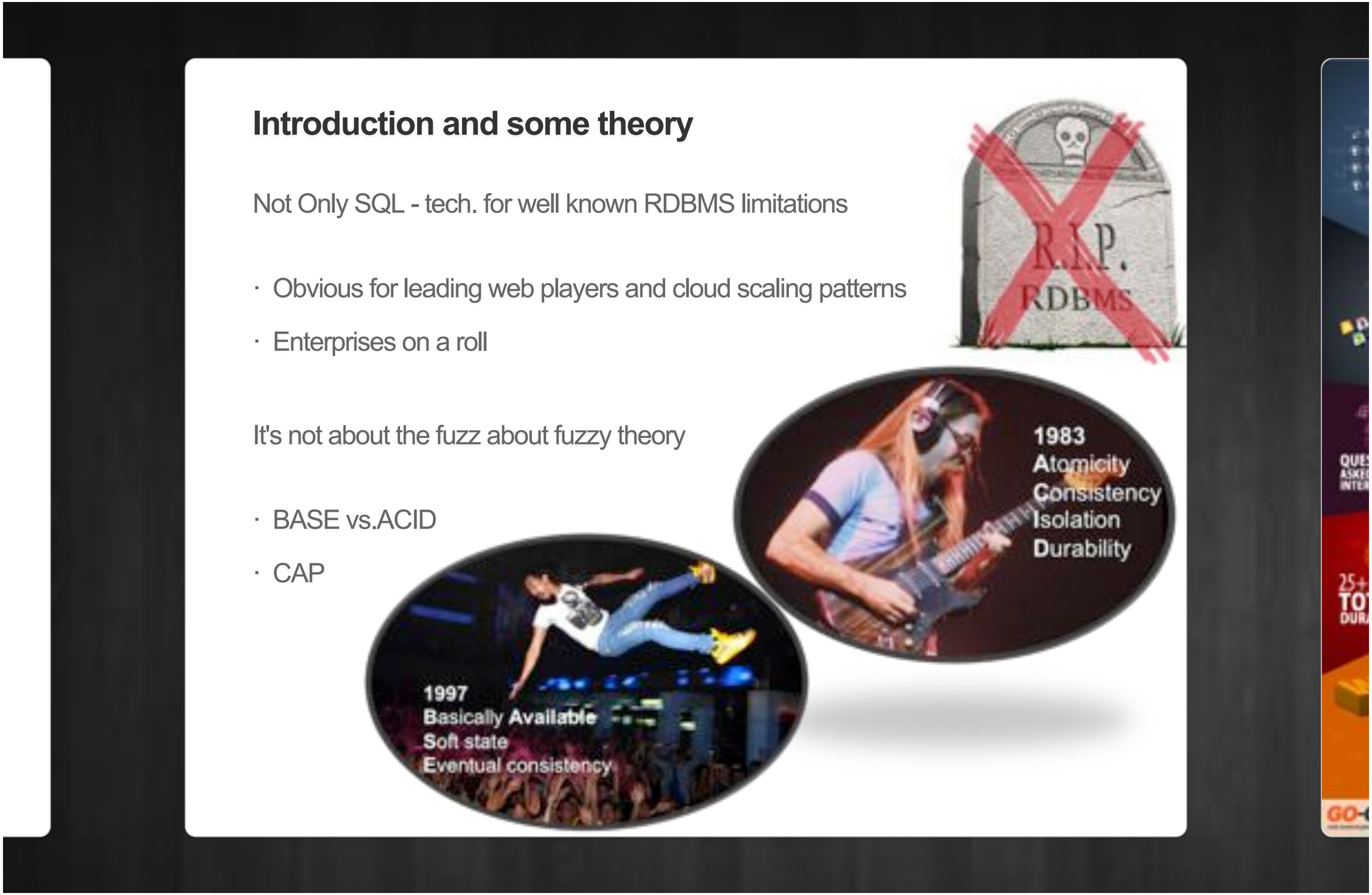
(SQL ⊆ NoSQL) != Oracle

peter.larsson@callistaenterprise.se

HOW TO WRITE A CV
DO YOU HAVE ANY EXPERTISE IN SQL?
NO
DOESN'T MATTER. WRITE: "EXPERT IN NO SQL"
Leverage the NoSQL boom

source: http://code.google.com/p/html5slides http://tinyurl.com/tlaunch



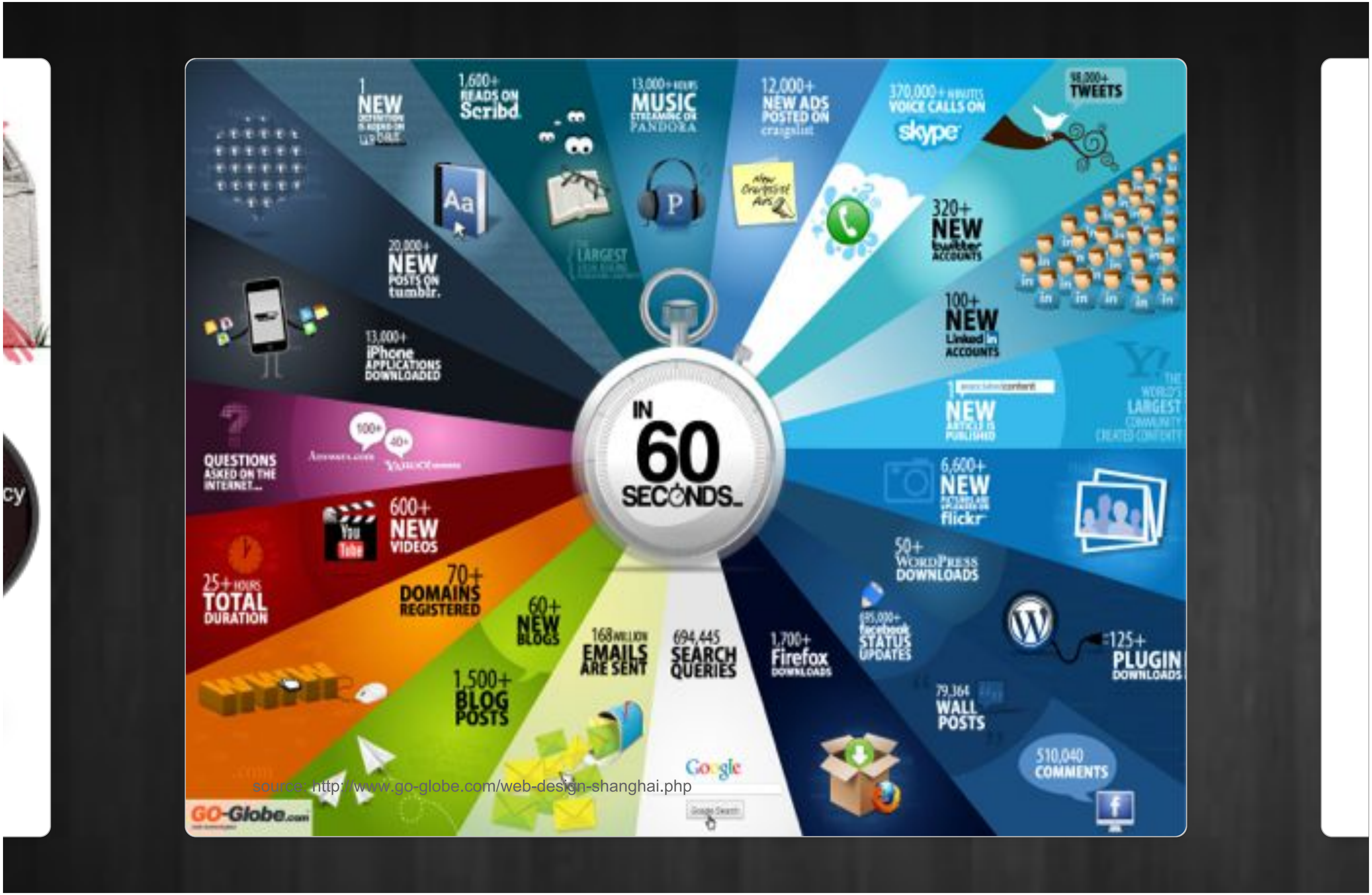
Introduction and some theory

Not Only SQL - tech. for well known RDBMS limitations

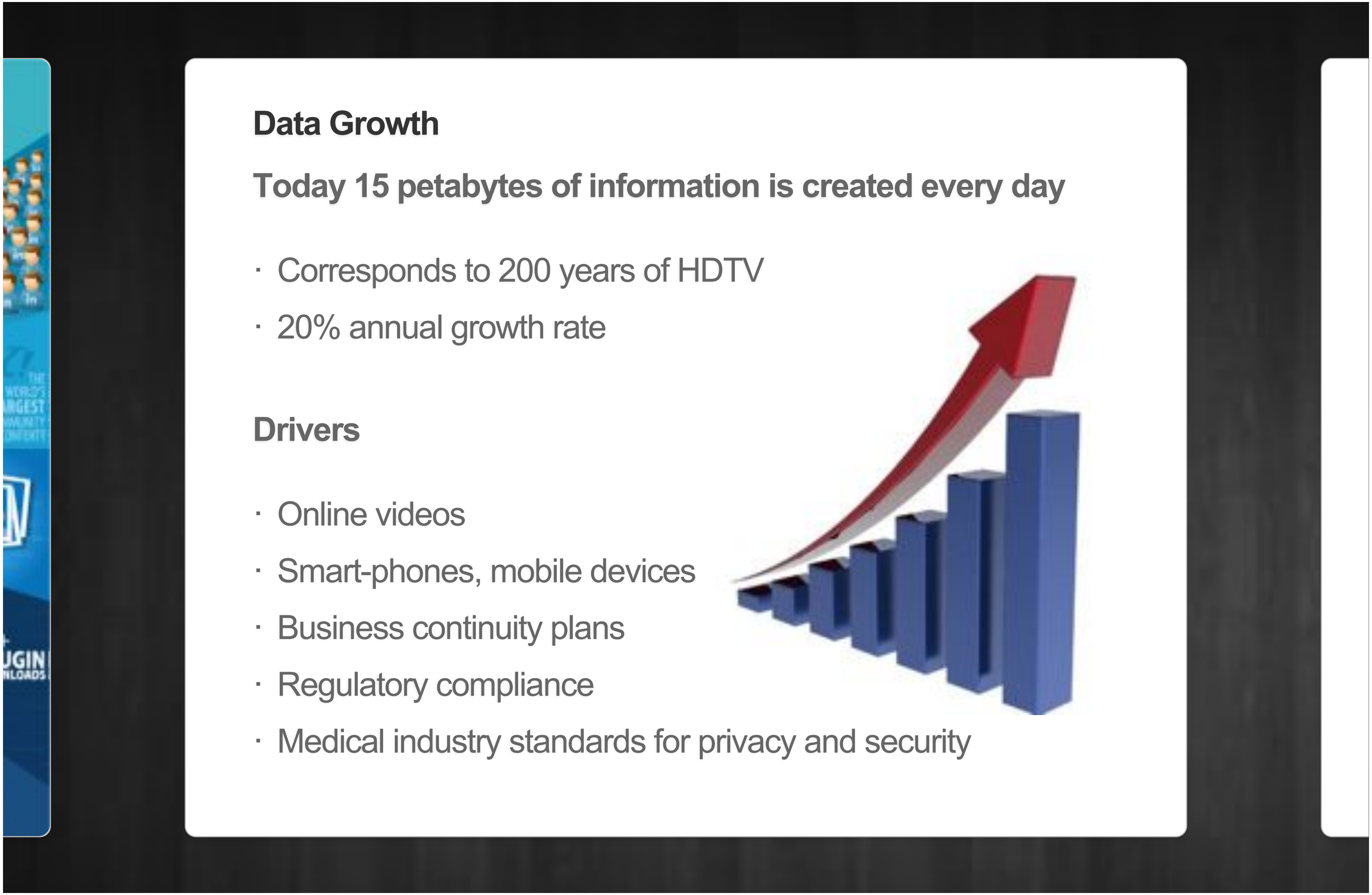
- Obvious for leading web players and cloud scaling patterns
- Enterprises on a roll

It's not about the fuzz about fuzzy theory

- BASE vs.ACID
- CAP



source: http://www.go-globe.com/web-design-shanghai.php



Data Growth

Today 15 petabytes of information is created every day

- Corresponds to 200 years of HDTV
- 20% annual growth rate

Drivers

- Online videos
- Smart-phones, mobile devices
- Business continuity plans
- Regulatory compliance
- Medical industry standards for privacy and security

Typical NoSQL Charateristics

At least from my point of view!

- Schema-free
- Easy cluster and replication setup
- Data partition tolerant
- Parallel processing, i.e. Map Reduce
- Language centric APIs over SQL
- Fault tolerant, commodity hardware over appliances

...DBA?

Types: Key-value, Document, Graph, and BigTable

Commodity vs. Appliance

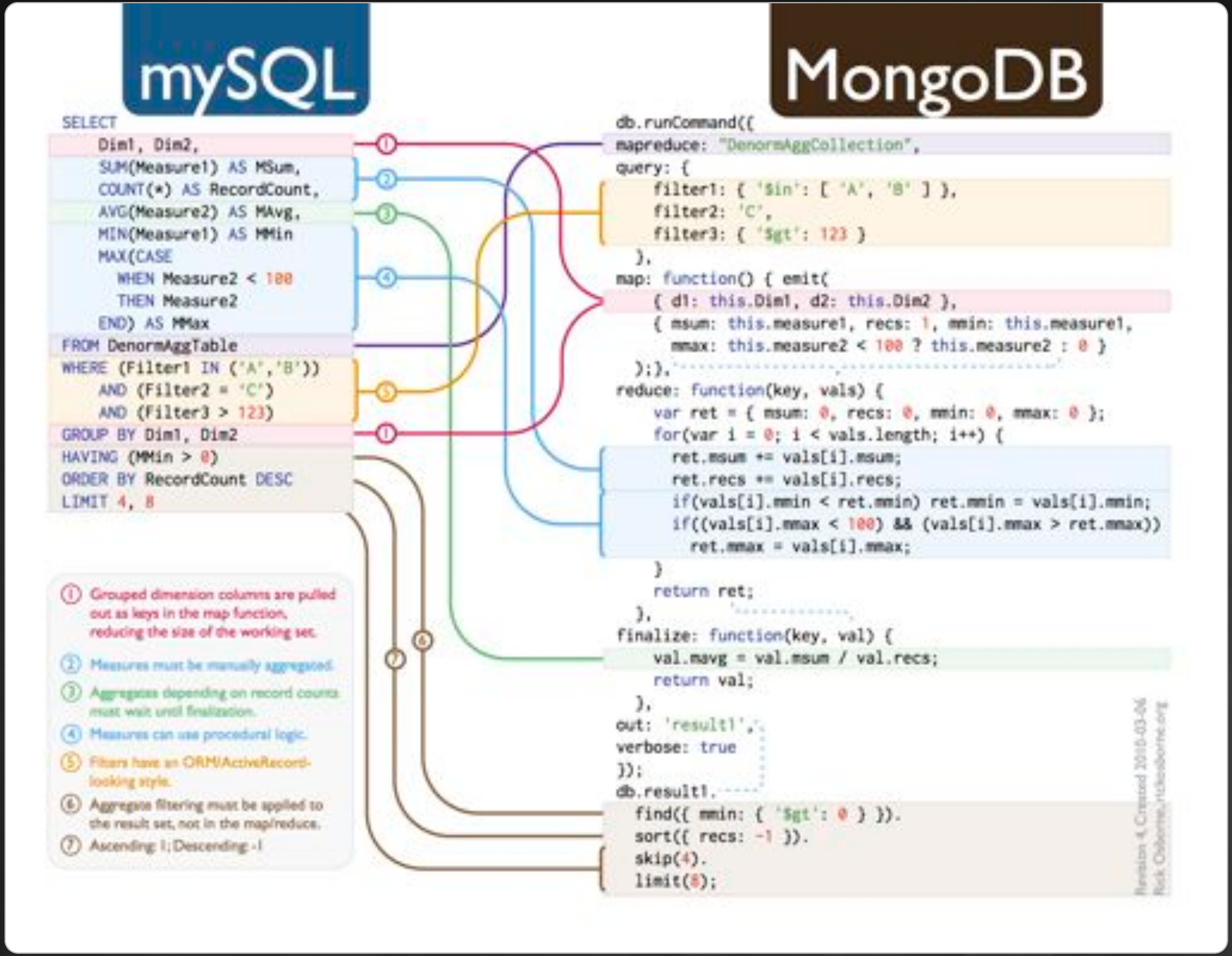
Google vs. Exadata

Map Reduce

So, how do I query the database?
It's not a database. It's a key-value store!
Ok, it's not a database. How do I query it?
You write a distributed map-reduce function in Erlang!
Did you just tell me to go fuck myself?
I believe I did, Bob!

...group by in SQL

source: http://browsertoolkit.com/fault-tolerance.png



```
SELECT
    Dim1, Dim2,
    SUM(Measure1) AS MSum,
    COUNT(*) AS RecordCount,
    AVG(Measure2) AS MAvg,
    MIN(Measure1) AS MMin
    MAX(CASE
      WHEN Measure2 < 100
      THEN Measure2
    END) AS MMax
FROM DenormAggTable
WHERE (Filter1 IN ('A','B'))
    AND (Filter2 = 'C')
    AND (Filter3 > 123)
GROUP BY Dim1, Dim2
HAVING (MMin > 0)
ORDER BY RecordCount DESC
LIMIT 4, 8
```

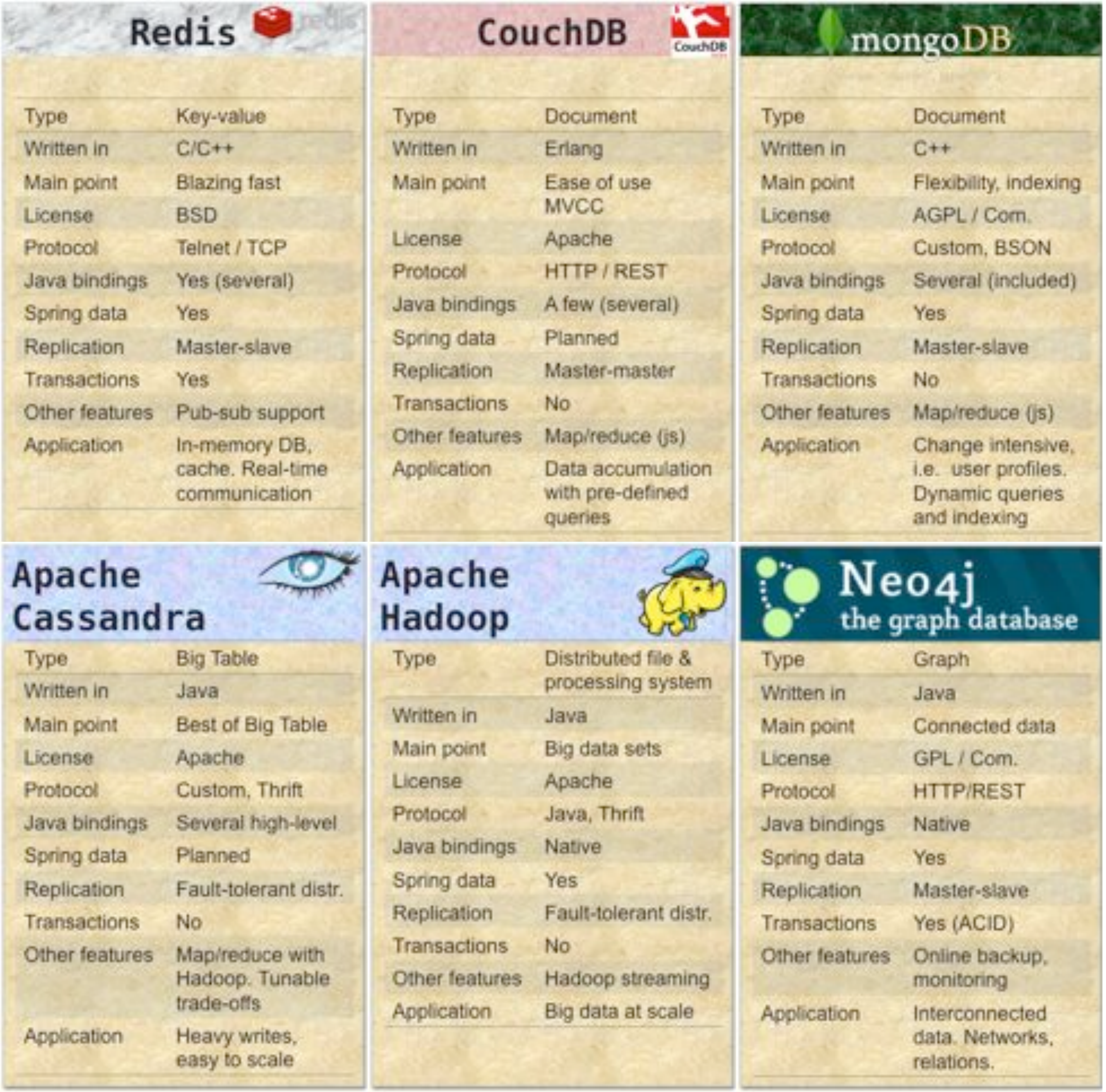
1. Grouped dimension columns are pulled out as keys in the map function, reducing the size of the working set.
2. Measures must be manually aggregated.
3. Aggregates depending on record counts must wait until finalization.
4. Measures can use procedural logic.
5. Filters have an ORM/ActiveRecord-looking style.
6. Aggregate filtering must be applied to the result set, not in the map/reduce.
7. Ascending: 1; Descending: -1

```
db.runCommand({
mapreduce: "DenormAggCollection",
query: {
    filter1: { '$in': [ 'A', 'B' ] },
    filter2: 'C',
    filter3: { '$gt': 123 }
  },
map: function() { emit(
    { d1: this.Dim1, d2: this.Dim2 },
    { msum: this.measure1, recs: 1, mmin: this.measure1,
      mmax: this.measure2 < 100 ? this.measure2 : 0 }
  );},
reduce: function(key, vals) {
    var ret = { msum: 0, recs: 0, mmin: 0, mmax: 0 };
    for(var i = 0; i < vals.length; i++) {
      ret.msum += vals[i].msum;
      ret.recs += vals[i].recs;
      if(vals[i].mmin < ret.mmin) ret.mmin = vals[i].mmin;
      if((vals[i].mmax < 100) && (vals[i].mmax > ret.mmax))
        ret.mmax = vals[i].mmax;
    }
    return ret;
  },
finalize: function(key, val) {
    val.mavg = val.msum / val.recs;
    return val;
  },
out: 'result1',
verbose: true
});
db.result1.
  find({ mmin: { '$gt': 0 } }).
  sort({ recs: -1 }).
  skip(4).
  limit(8);
```

Revision 4, Created 2010-03-06
Rick Osborne, rickosborne.org

Should you care?

source: http://epicponyz.files.wordpress.com/2009/05/confused-and-quizical.jpg



Redis

Type	Key-value
Written in	C/C++
Main point	Blazing fast
License	BSD
Protocol	Telnet / TCP
Java bindings	Yes (several)
Spring data	Yes
Replication	Master-slave
Transactions	Yes
Other features	Pub-sub support
Application	In-memory DB, cache. Real-time communication

CouchDB

Type	Document
Written in	Erlang
Main point	Ease of use MVCC
License	Apache
Protocol	HTTP / REST
Java bindings	A few (several)
Spring data	Planned
Replication	Master-master
Transactions	No
Other features	Map/reduce (js)
Application	Data accumulation with pre-defined queries

mongoDB

Type	Document
Written in	C++
Main point	Flexibility, indexing
License	AGPL / Com.
Protocol	Custom, BSON
Java bindings	Several (included)
Spring data	Yes
Replication	Master-slave
Transactions	No
Other features	Map/reduce (js)
Application	Change intensive, i.e. user profiles. Dynamic queries and indexing

Apache Cassandra

Type	Big Table
Written in	Java
Main point	Best of Big Table
License	Apache
Protocol	Custom, Thrift
Java bindings	Several high-level
Spring data	Planned
Replication	Fault-tolerant distr.
Transactions	No
Other features	Map/reduce with Hadoop. Tunable trade-offs
Application	Heavy writes, easy to scale

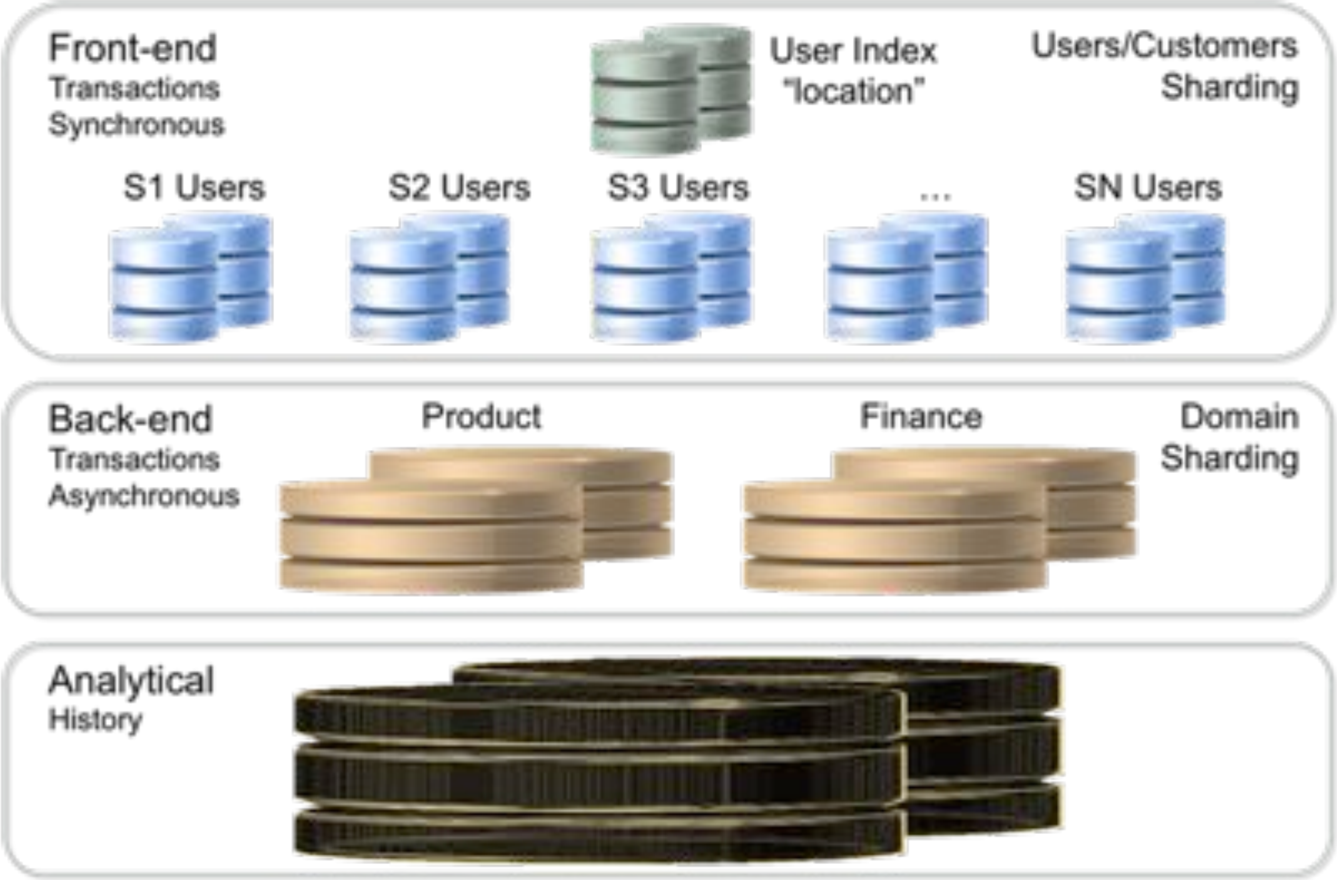
Apache Hadoop

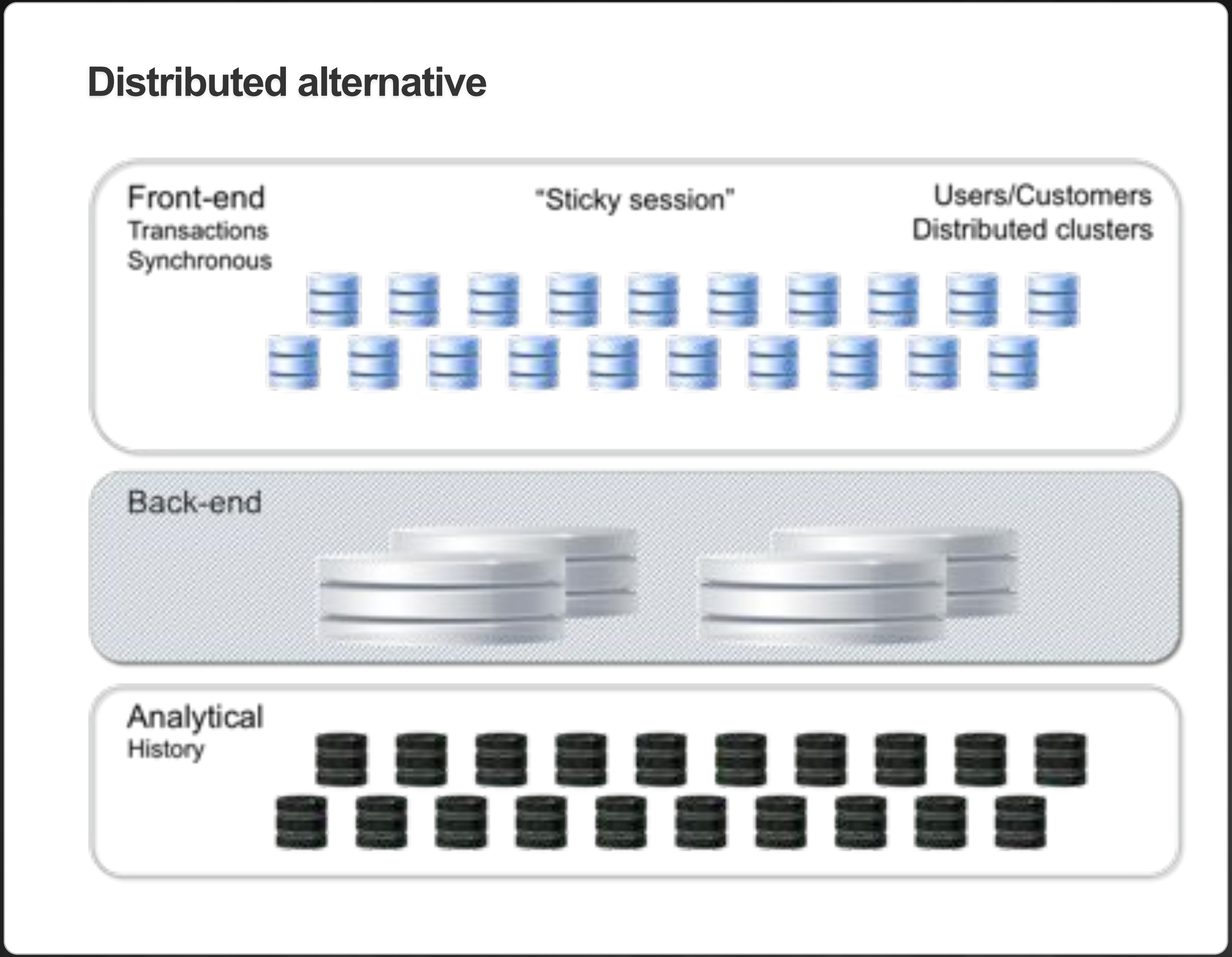
Type	Distributed file & processing system
Written in	Java
Main point	Big data sets
License	Apache
Protocol	Java, Thrift
Java bindings	Native
Spring data	Yes
Replication	Fault-tolerant distr.
Transactions	No
Other features	Hadoop streaming
Application	Big data at scale

Neo4j the graph database

Type	Graph
Written in	Java
Main point	Connected data
License	GPL / Com.
Protocol	HTTP/REST
Java bindings	Native
Spring data	Yes
Replication	Master-slave
Transactions	Yes (ACID)
Other features	Online backup, monitoring
Application	Interconnected data. Networks, relations.

A typical RDBMS landscape, sharding





Distributed alternative

Front-end
Transactions
Synchronous

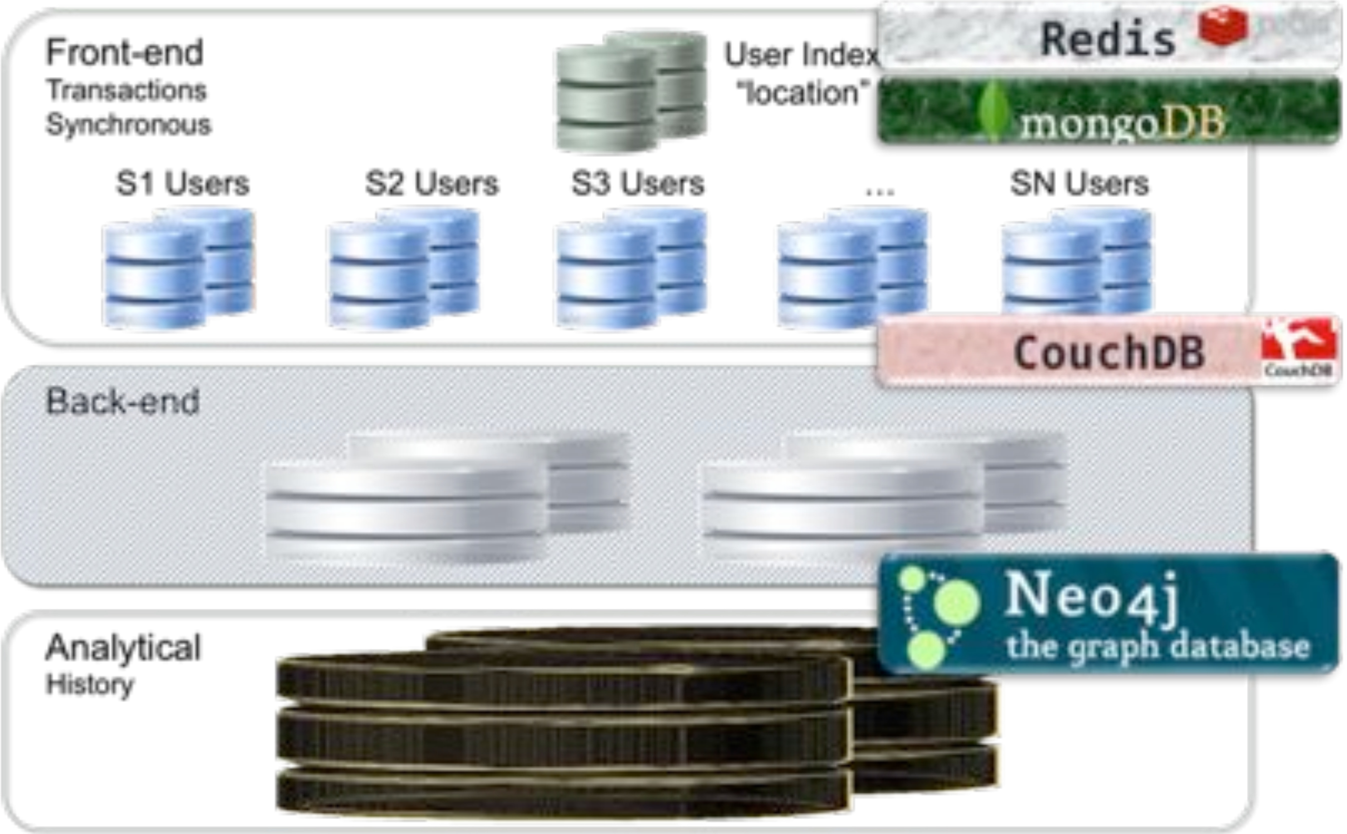
"Sticky session"

Users/Customers
Distributed clusters

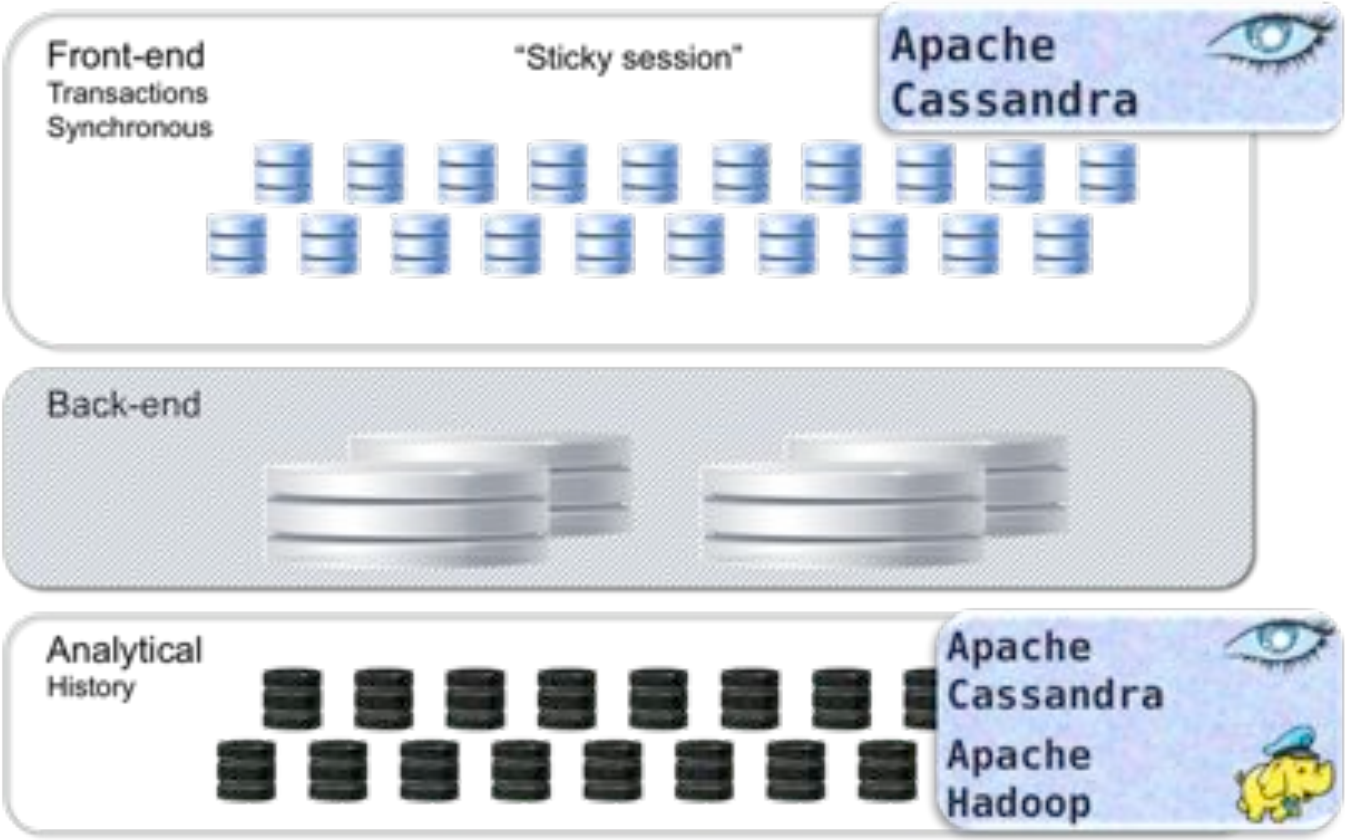
Back-end

Analytical
History

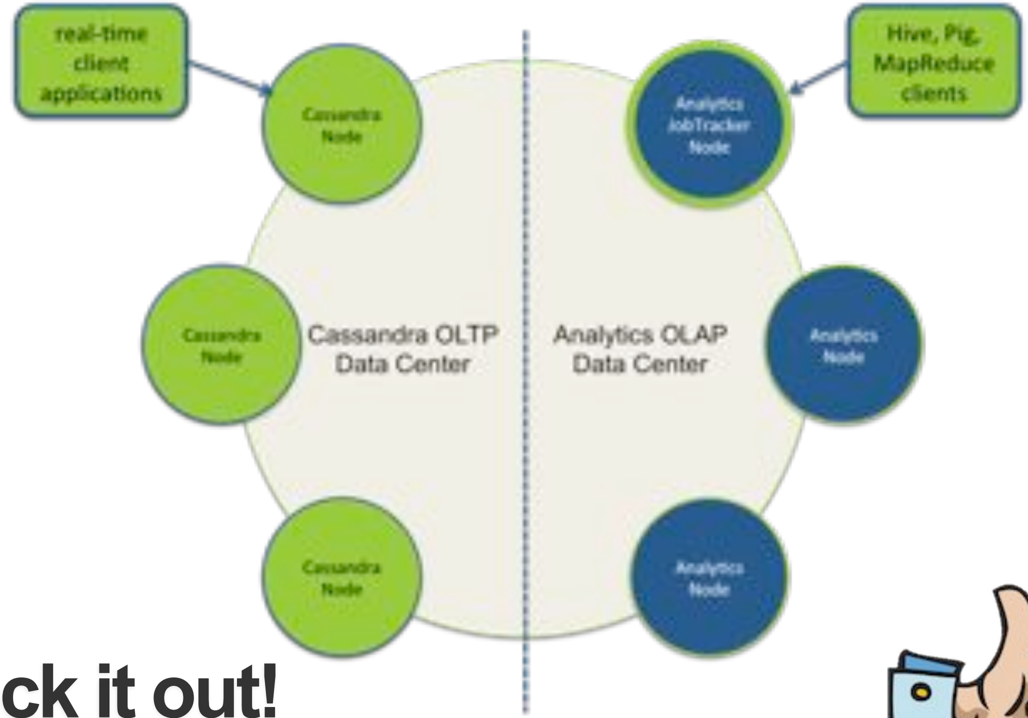
Just like any RDBMS are...



Apache makes a difference...



DataStax - Hadoop and Cassandra combined!


real-time client applications
Cassandra Node
Cassandra Node
Cassandra OLTP Data Center
Cassandra Node
Hive, Pig, MapReduce clients
Analytics JobTracker Node
Analytics OLAP Data Center
Analytics Node
Analytics Node

Check it out!

http://www.datastax.com
http://www.nosqldatabases.com/main/2011/3/2/how-to-setup-a-cassandra-cluster-in-2-minutes.html

That's all!

	⬣⬣
	⬣⬣

...and the winner is?