copenhagen goto; conference

TALK 1:

CONVINCE YOUR BOSS:

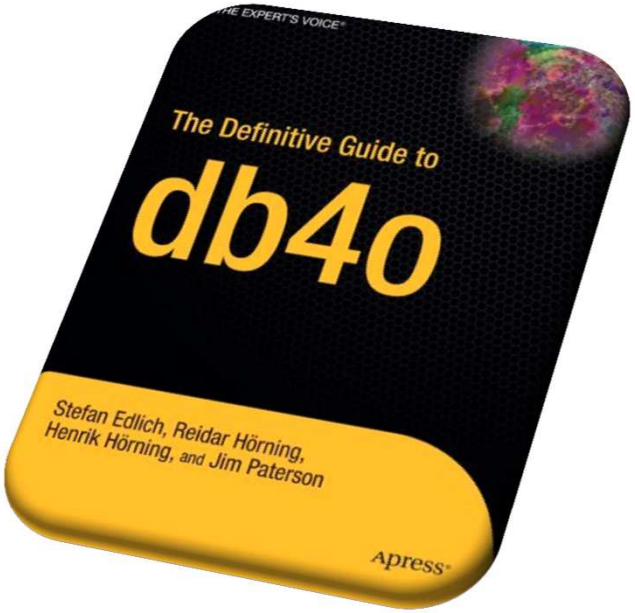
CHOOSE THE "RIGHT" DATABASE

Prof. Dr. Stefan Edlich

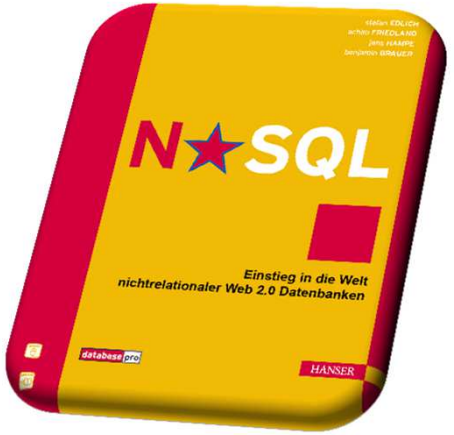
Beuth University of Technology Berlin (App.Sc.)

INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

gotocon.com


THE EXPERT'S VOICE®
The Definitive Guide to
db4o
Stefan Edlich, Reidar Hörning,
Henrik Hörning, and Jim Paterson
Apress®


Urheberrechtlich geschütztes Material
stefan EDLICH
achim FRIEDLAND
jens HAMPE
benjamin BRAUER
markus BRÜCKNER
NoSQL
2. Auflage
EINSTIEG IN DIE WELT
NICHTRELATIONALER
WEB 2.0 DATENBANKEN
E-Book
EXTRA: Mit kostenlosem E-Book
NEU: Jetzt mit Kapiteln zu OrientDB
und Membase
HANSER
Urheberrechtlich geschütztes Material


NoSQL
Einstieg in die Welt
nichtrelationaler Web 2.0 Datenbanken
HANSER

N★SQL Berlin

peritor consulting · studiVZ · VERSANT · sociomantic labs · SOUNDCLOUD

Thursday, October 22nd, 15:00,
Newthinking Store, Berlin, Germany

View Larger Map

The first NoSQL Meetup Germany was located in Berlin, October 22nd 2009 – 15:00. The event comprised presentations on open source, distributed, non relational databases.

Schedule

- 15:00 — Opening, Welcome, Intro
- 15:15 — Consistency in Key-Value Stores, *Monika Moser* (Video, Slides)
- 15:45 — Redis, Fast and Furious, *Mathias Meyer* (Video, Slides)
- 16:15 — Questions & Discussion
- 16:25 — Short Break
- 16:35 — Peer-to-peer Applications with CouchDB, *Jan Lehnardt* (Video, Slides)
- 17:05 — Riak, *Martin Scholl* (Video, Slides)
- 17:35 — Questions & Discussions
- 17:45 — Short Break
- 17:55 — MongoDB, *Mathias Stearn* (Video, Slides)
- 18:25 — 4th Generation Object Databases, *Prof. Stefan Edlich* (Video, Slides)
- 18:55 — Closing Discussion
- 19:15 — Dinner & Drinks at Aufsturz

icoodb 2010

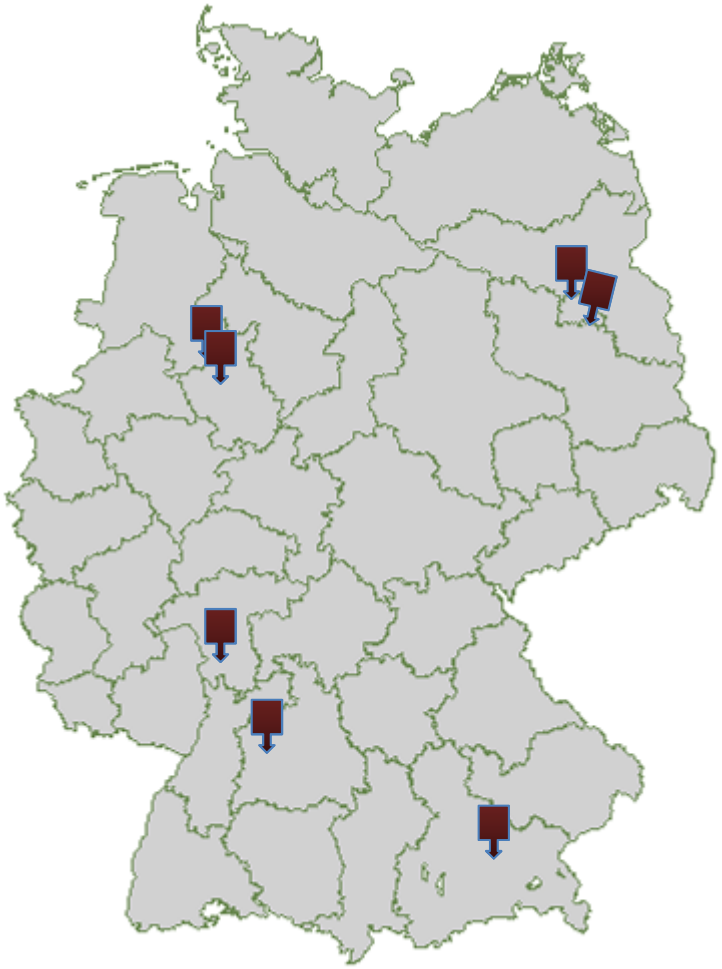
nosqlfrankfurt.de
nosql powerdays
...






N★SQL matters

2 years of NoSQL Consulting



HTTP://NOSQL-DATABASE.ORG

NoSQL

Your Ultimate Guide to the Non - Relational Universe!

[the **biggest** nosql link Archive in the web]
...never miss a conceptual article again...
News Feed covering all changes here !

NoSQL DEFINITION: Next Generation Databases mostly addressing some of the points: being **non-relational, distributed, open-source** and **horizontal scalable**. The original intention has been **modern web-scale databases**. The movement began early 2009 and is growing rapidly. Often more characteristics apply as: **schema-free, easy replication support, simple API, eventually consistent** / **BASE** (not ACID), a **huge data amount**, and more. So the misleading term *"nosql"* (the community now translates it mostly with "**not only sql**") should be seen as an alias to something like the definition above.
[based on 5 sources, 10 constructive feedback emails (thanks!) and1 insulting comment. Agree / Disagree? Tell me so!]

LIST OF NOSQL DATABASES [beta]

Core NoSQL Systems:

Wide Column Store / Column Families

Hadoop / HBase: API: **Java / any writer**, Protocol: **any write call**, Query Method: **MapReduce Java / any exec**, Replication: **HDFS Replication**, Written in: **Java**, Concurrency: ?, Misc: **Links**: 3 Books [1, 2, 3]

Cassandra: API: **many Thrift » languages**, Protocol: ?, Query Method: **MapReduce**, Replicaton: , Written in: **Java**, Concurrency: **eventually consistent** , Misc: like "Big-Table on Amazon Dynamo alike", initiated by Facebook, Slides » , Clients », Installation »

Hypertable: API: **Thrift** (Java, PHP, Perl, Python, Ruby, etc.), Protocol: **Thrift**, Query Method: **HQL, native Thrift API**, Replication: **HDFS Replication**, Concurrency: **MVCC**, Consistency Model: **Fully consistent** Misc: High performance C++ implementation of Google's Bigtable. Commercial support »

Cloudera: Professional Software & Services based on Hadoop.

Amazon SimpleDB: **Misc**: not open source / part of AWS, Book »

[SciDB: **Array** Data Model for Scientists, paper », poster », HiScaBlog »]
[OpenNeptune, Qbase, KDI]:

EVENTS

17th JAN **Membase** Seattle »
5th JAN **Membase** San Diego
3rd DEC **Mongo** Mountain View
20th NOV NoSQL **Dundee** »
All past NoSQL Conferences »
register your event here! »

NoSQL Summer in 27 cities »

Conceptual quality articles, links, research papers, etc.

NoSQL

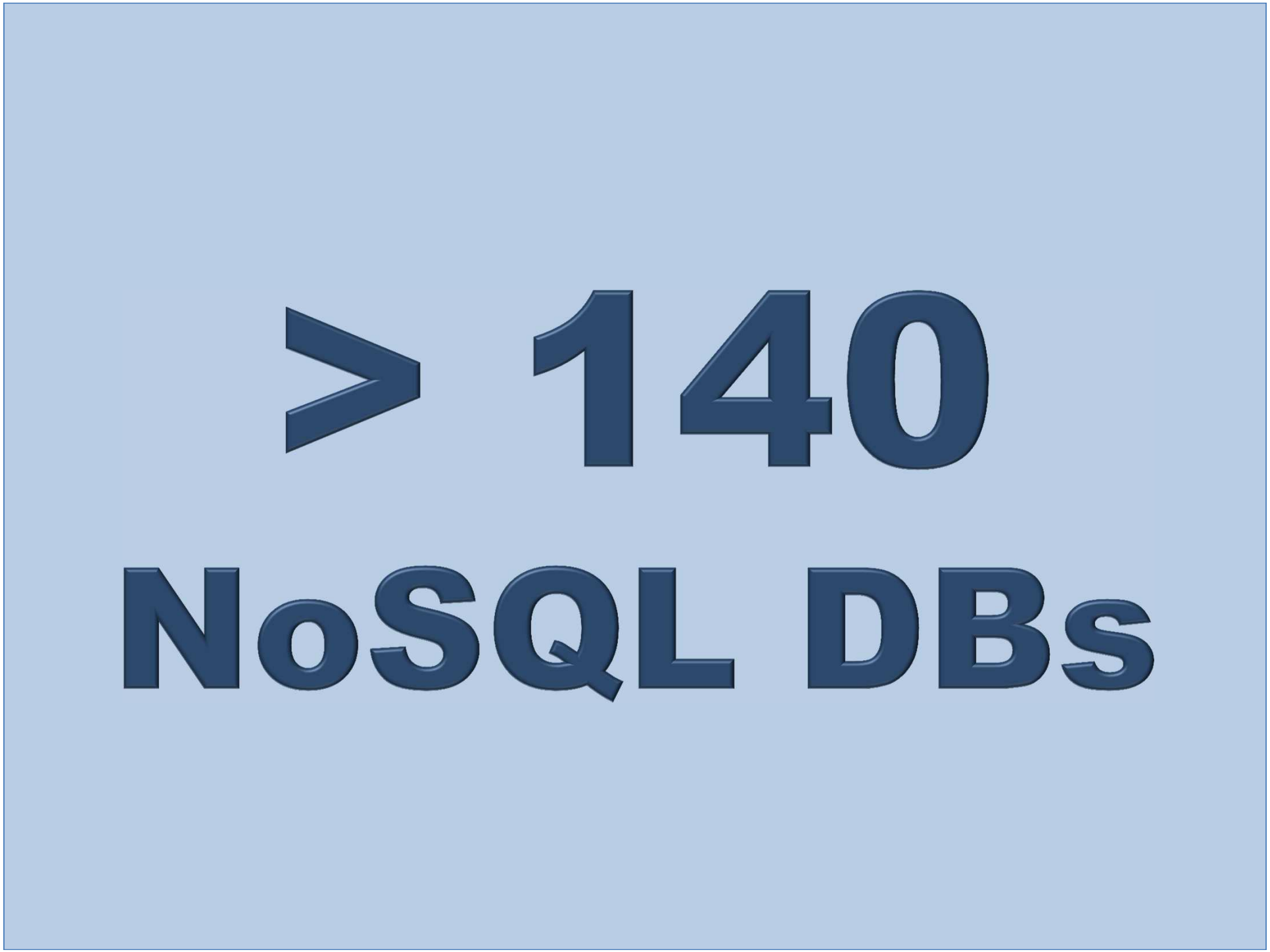
ARCHIVE

Announcing the availability of the **worlds first NoSQL Book**!

NoSQL FORUMS [3]
* Global NOSQL Forum »
* Forum Berlin »
* Forum France »
* Forum Japan »

GREAT NoSQL NEWS FEEDS
* MyNoSQL by Alex P »
[He is definetely bigger...]
* On Twitter: nosqlupdate »
* HighScalability Blog »

BREAKING NoSQL NEWS


> 140
NoSQL DBs

1st idea

+ Scaling= new node
+ Community
+ API
- Replication
- SetUp, Optimize, Management

+ Scaling= new node
+ Replication
+ Configuration (r, w)
- Documentation
- Query
- management

2nd idea

compare!

	HBase	Cassand.	DynDB	Mongo	CouchBS	Riak	Redis	ES
schema free	Chunks	TAB	TAB	JSON	JSON	JSON	K/V	JSON
‚realtime‘								
performance	mass data			100k			100k	
scaling							V3	
ring / shardRepl	chunks	ring	invisible	ShRe	ring	ring	repl	ShRe
self tuning		In progress	SaaS	hard				
prod / tools			why?	MMS			Why?	
aggregations						?	DSS	
queries	Hive,Pig	CQL?					DSS	
full text s		?					DSS	
filemanagem								
community								
APIs				++	-		++	REST
support			why?					
docs				++	+/-			
misc				Geo, StoredProcs	Geo	persist config	limited sharding	persist config

Best approach


Analyze-Data
non-funct-Requirements
Transactions-Consistency
Queries
Performance
Architecture

Now it‘s you!
And your company!

1A ⇨ Data

Message-Data

Log-Data

Event-Data

Session Data

Domain-Data

Meta-Data

Critical Data

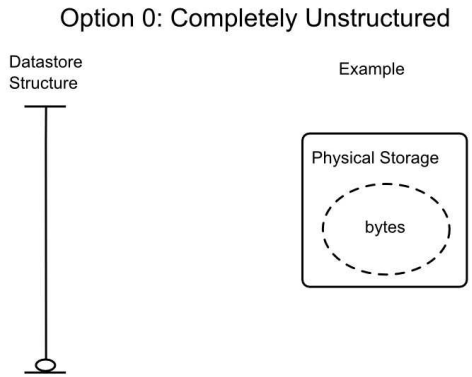
Business-Data

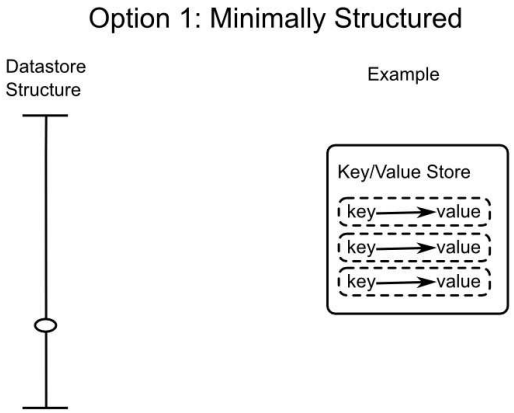
Geo Data

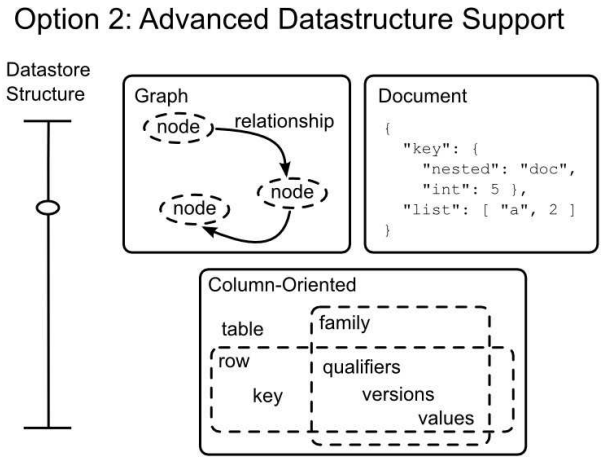
Temporal Data

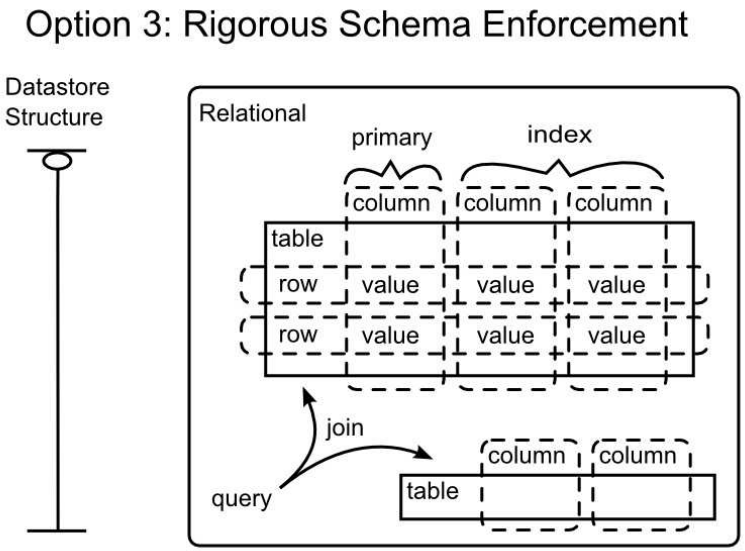
Cluster

1B ⇨ Storage Model







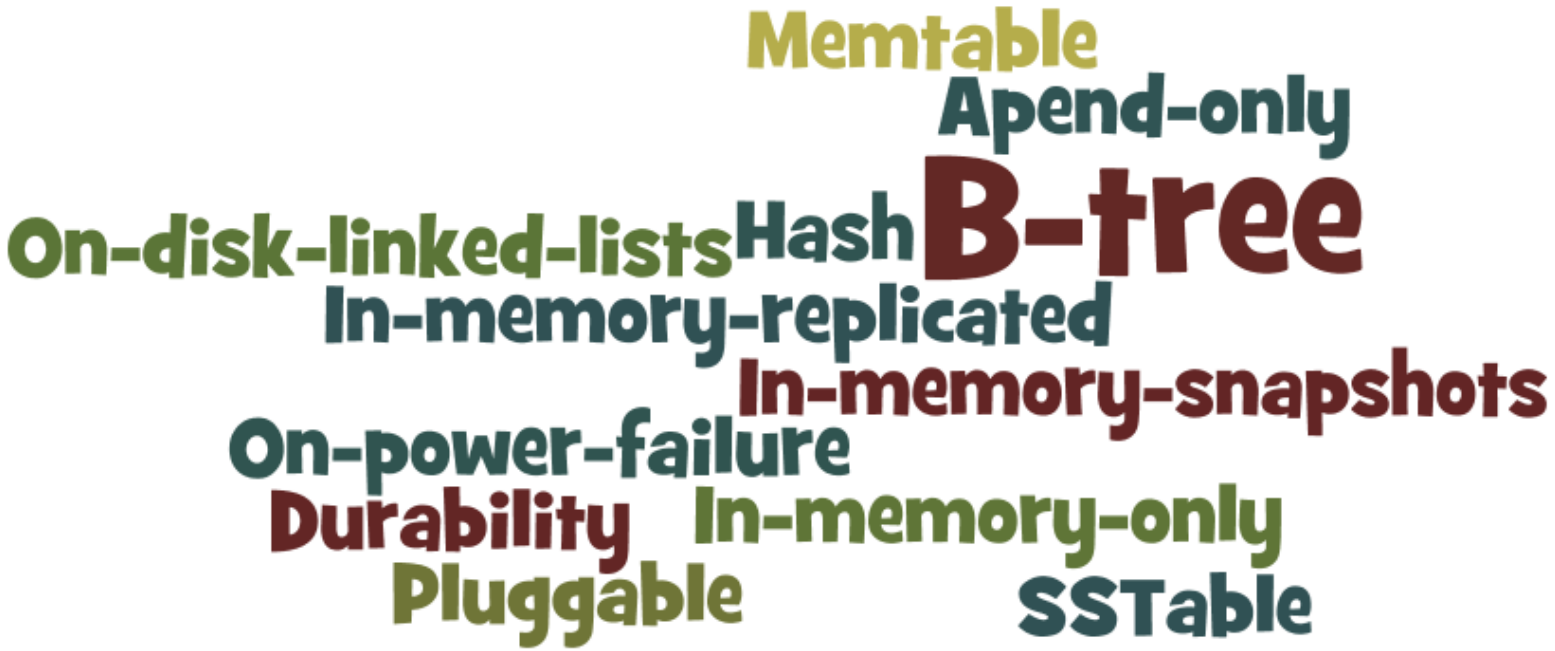


model - query trap

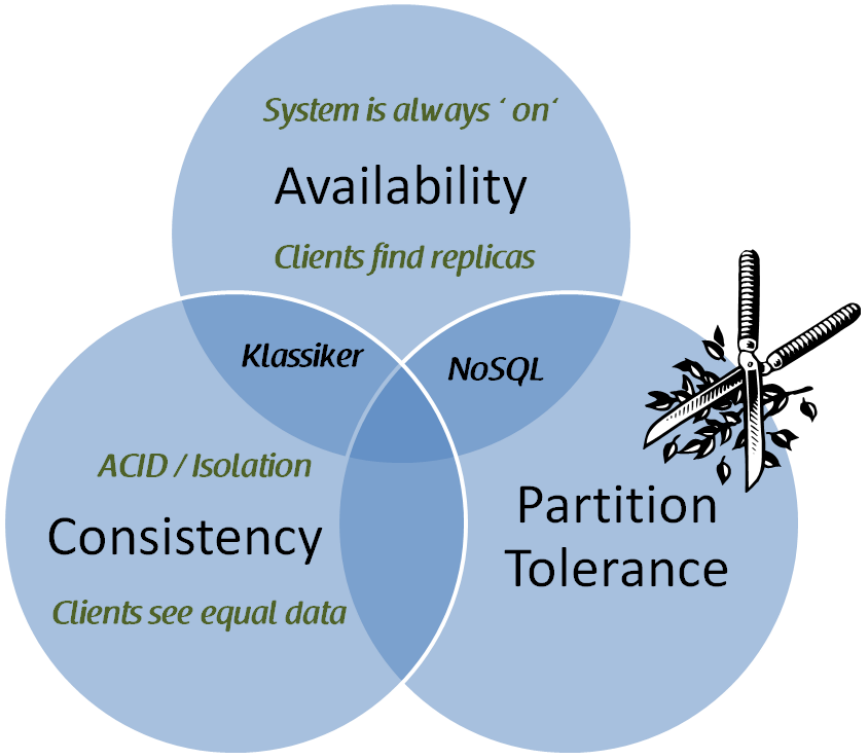
http://pragprog.com/magazines/2012-05/beyond-the-bit-bucket

1C ⇨ Data / Type constraints

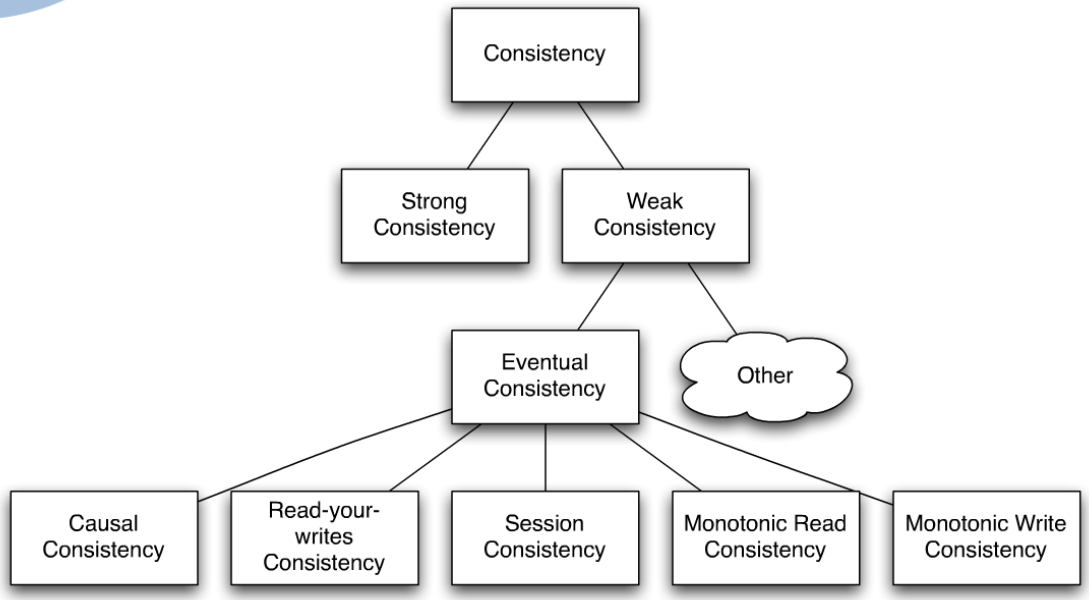
1D ⇨ Persistence Design



With input from highscalability.com



Lessons learned: Customer: "ACID!"



Latency & Request behaviour / distribution?

Throughput?

High Concurrency?

Lessons learned: model forces queries

Typical Queries look like?
SQL needed? LINQ needed?
BI / Analytic-Tools needed?
MapReduce needed?
Ad-Hoc Queries needed?
Background Data Analytics?
Secondary Indices?
Range queries?
Comlpex Aggregations?
ColumnDB needed for Analytics?
Views needed?

Architecture :

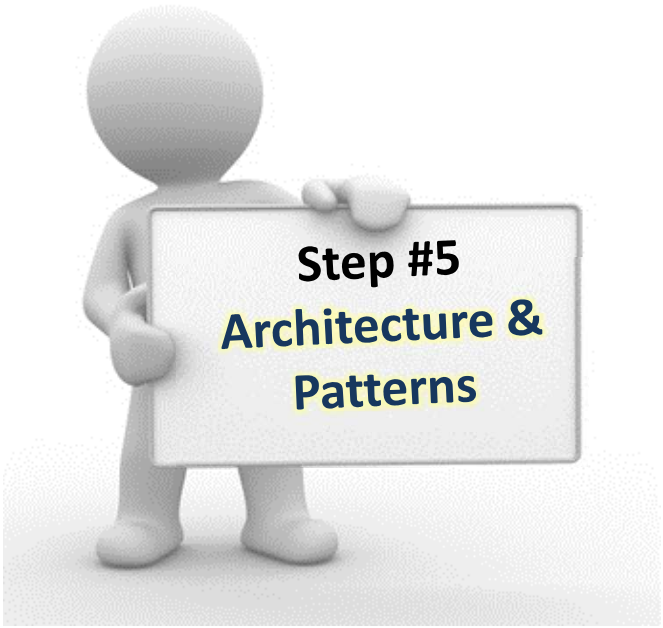
- local, parallel, distributed / grid, service, cloud, mobile, p2p, …
- Hosted? Cloud? Local? Datacenter?

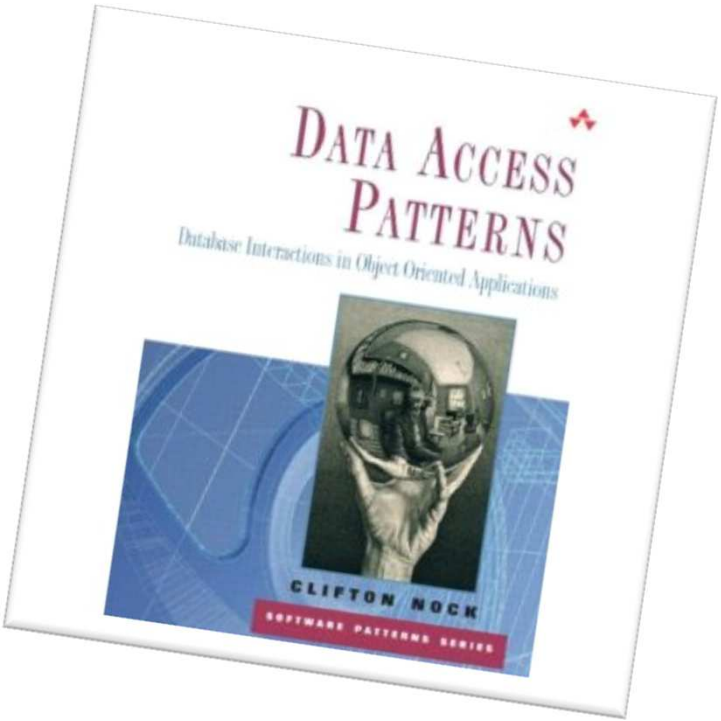
Data Access Patterns

- read / write distribution?
- random / sequential access?
- Access Design Patterns?

Lessons learned:

- **SSD vs. Disk**
- **write master + read slaves**
- **…**





Lessons learned:
Should be checked first!
They put you out of the game!

Integration / Tools_

- Replication needed? = Rubustness
- Automatic load balancing, partitioning, and repartitioning?
- Auto-Scaling needed?
- Text search integration? Lucene / Solr?

Real non-functional Requirements:

- Refactoring Frequency?
- 24/7 System? Live add and remove?
- Developer Qualification
- DB simplicity? (installation, configuration, development, deployment, upgrade)
- Company restrictions?
- DB diversity (allowed?)
- Security? (authentication, authorization, validation?)
- Licence Model?
- Vendor trustworthiness?
- Community support?
- Documentation?
- Company and DB dev in the future?

Costs:

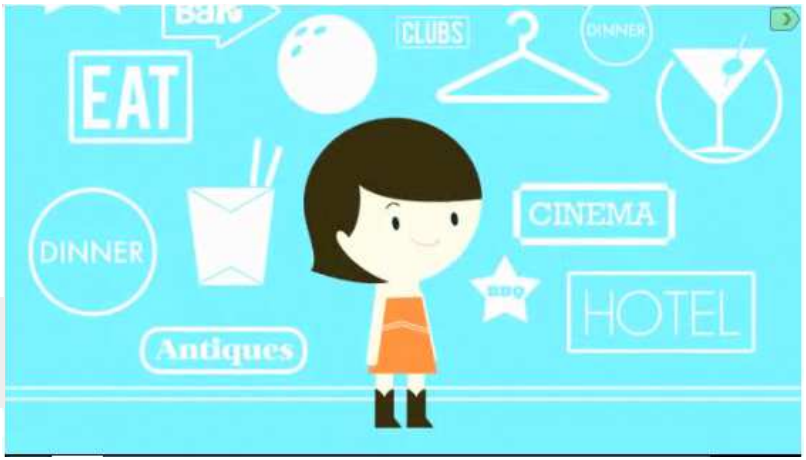
- DB-Support? (responsiveness, SLA)
- Costs in general, Scaling Costs
- Sysadmin costs

Operational Costs: (noOps)

- Safety / Backup & Restore
- Crash Resistance, Disaster Management
- Monitoring

Failed Ops

foursquare



67 GB
Index
Data ✓

EC2 Node 66 GB

EC2 Node 66 GB

11 hours + 1 day off

```
connected to: 127.0.0.1
insert  query update delete getmore command flushes mapped  vsize    res locked % idx miss %     qr|qw   ar|aw   netIn netOut  conn       time
     0      0      0      0       0       1       0     0m  2.36g     4m        0          0       0|0     0|0     62b     1k     1   17:25:35
     0      0      0      0       0       1       0     0m  2.36g     4m        0          0       0|0     0|0     62b     1k     1   17:25:36
     0      0      0      0       0       1       0     0m  2.36g     4m        0          0       0|0     0|0     62b     1k     1   17:25:37
     0      0      0      0       0       1       0     0m  2.36g     4m        0          0       0|0     0|0     62b     1k     1   17:25:38
```

Good Ops

mongod Edlich-PC

List all commands | Replica set status

Commands: buildInfo cursorInfo features isMaster listDatabases replSetGetStatus serve

```
db version v1.8.2, pdfile version 4.5
git hash: 433bbaa14aaba6860da15bd4de8edf600f56501b
sys info: windows (6, 1, 7600, 2, '') BOOST_LIB_VERSION=1_42
uptime: 8 seconds
```

low level requires read lock

```
time to get readlock: 0ms
# databases: 1

replication:
master: 0
slave:  0
initialSyncCompleted: 1
```

clients

Client	Opid	Active	LockType	Waiting	SecsRunning	Op	Namespa
websvr	0		0			0	
snapshotthread	0		0			0	
initandlisten	0		W			2004	test
clientcursormon	0		R			0	
			(NONE)				

dbtop (occurences|percent of elapsed)

NS	total		Reads		Writes		Queries		GetMores		Inserts		Updates		Removes	
TOTAL	5	3.2%	1	0%	4	3.2%	5	3.2%	0	0%	0	0%	0	0%	0	0%
comments	1	0.6%	0	0%	1	0.6%	1	0.6%	0	0%	0	0%	0	0%	0	0%
edltest	1	0.7%	0	0%	1	0.7%	1	0.7%	0	0%	0	0%	0	0%	0	0%
local.system.namespaces	1	0%	1	0%	0	0%	1	0%	0	0%	0	0%	0	0%	0	0%
privat	1	1.1%	0	0%	1	1.1%	1	1.1%	0	0%	0	0%	0	0%	0	0%
test	1	0.8%	0	0%	1	0.8%	1	0.8%	0	0%	0	0%	0	0%	0	0%

write lock % time in write lock, by 4 sec periods
0 0
write locked now: false

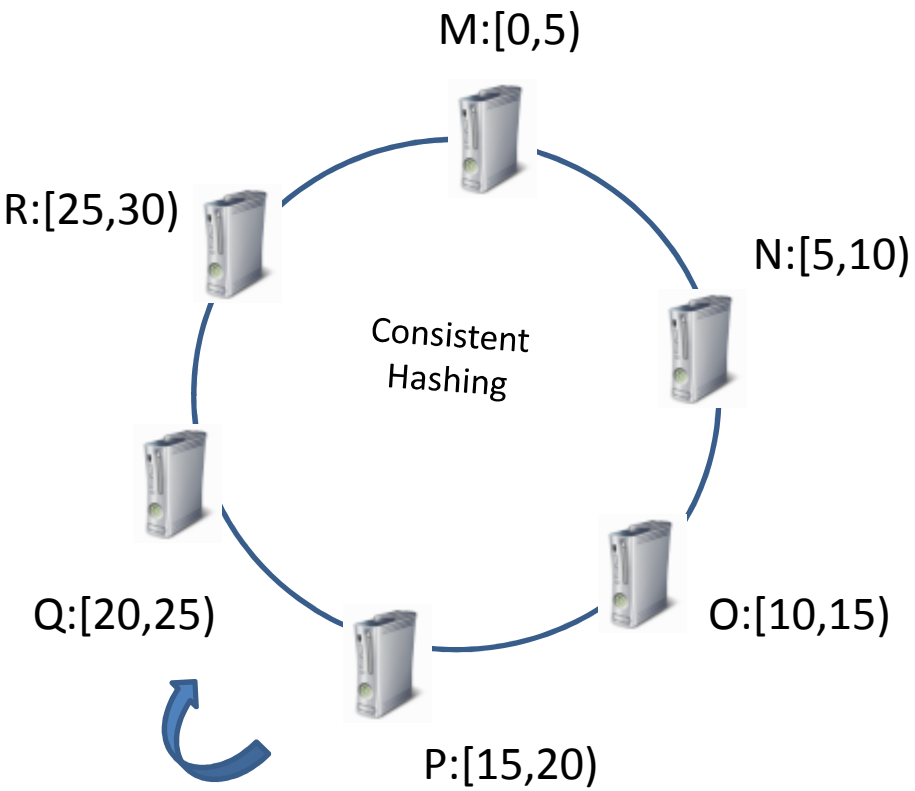
Log

```
Fri Jul 01 14:28:19 BackgroundJob starting: ClientCursorMonitor
           14:28:19 [websvr] web admin interface listening on port 28017
```

db.server.status() !!
Chef, Puppet, etc.
MaaS: ServerDensity

Better Ops

Andy Gross: „Operational costs trump everything!“



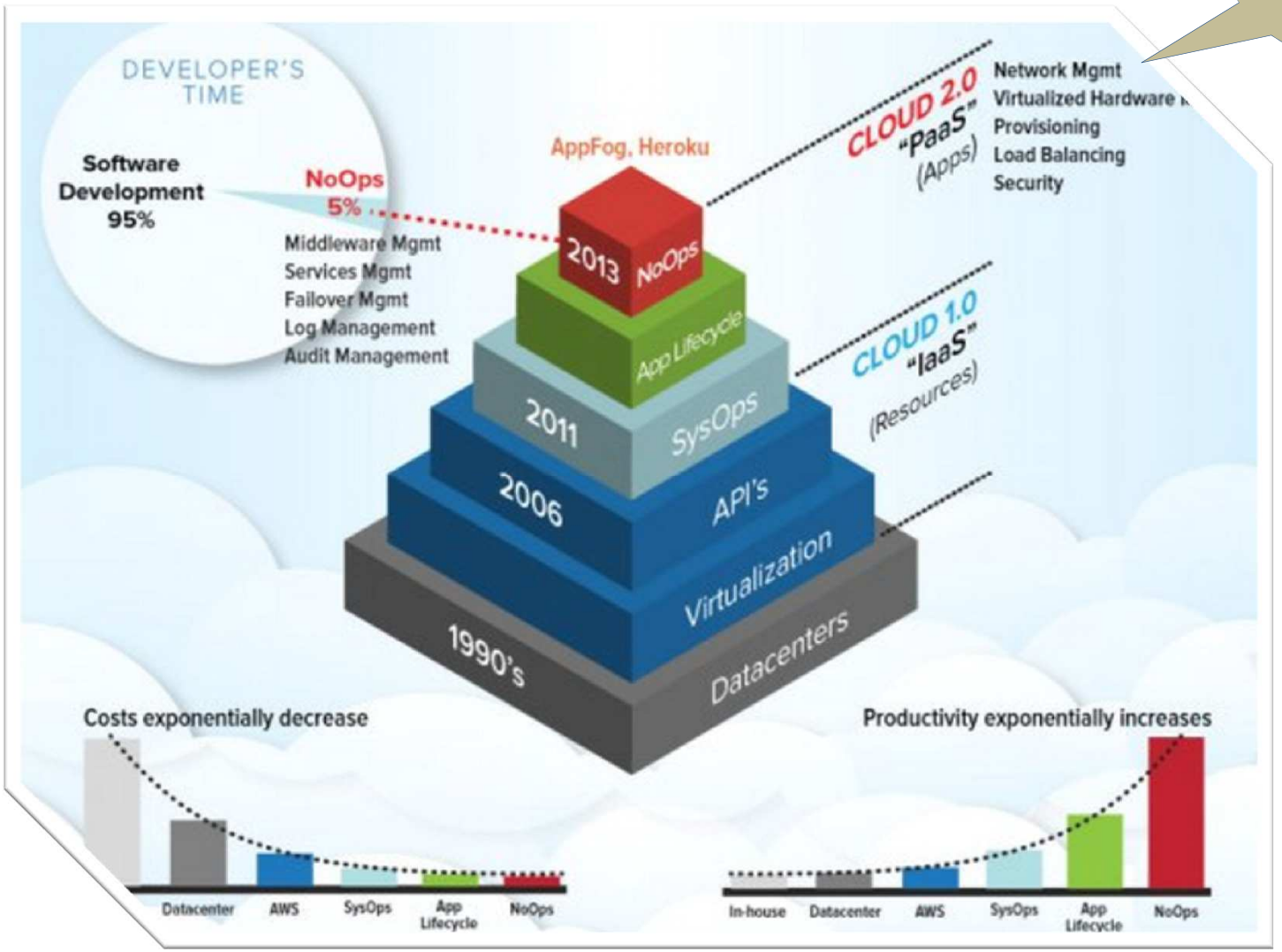
HASH	KNOTEN	REPLIKAT
2	M	N,O
8	N	O,P
10	O	P,Q
17	P	Q,R
22	Q	R,M
26	R	M,N

*W ✓ = 2*W*
*R ✓ = 1*R*

- *fail save ✓*
- *easy extensible ✓*
- *perfectly distributed / vnodes ✓*

No Ops

With NoSQL!



Guess the top 3 blockers for NoSQl or Polyglot Persistence?

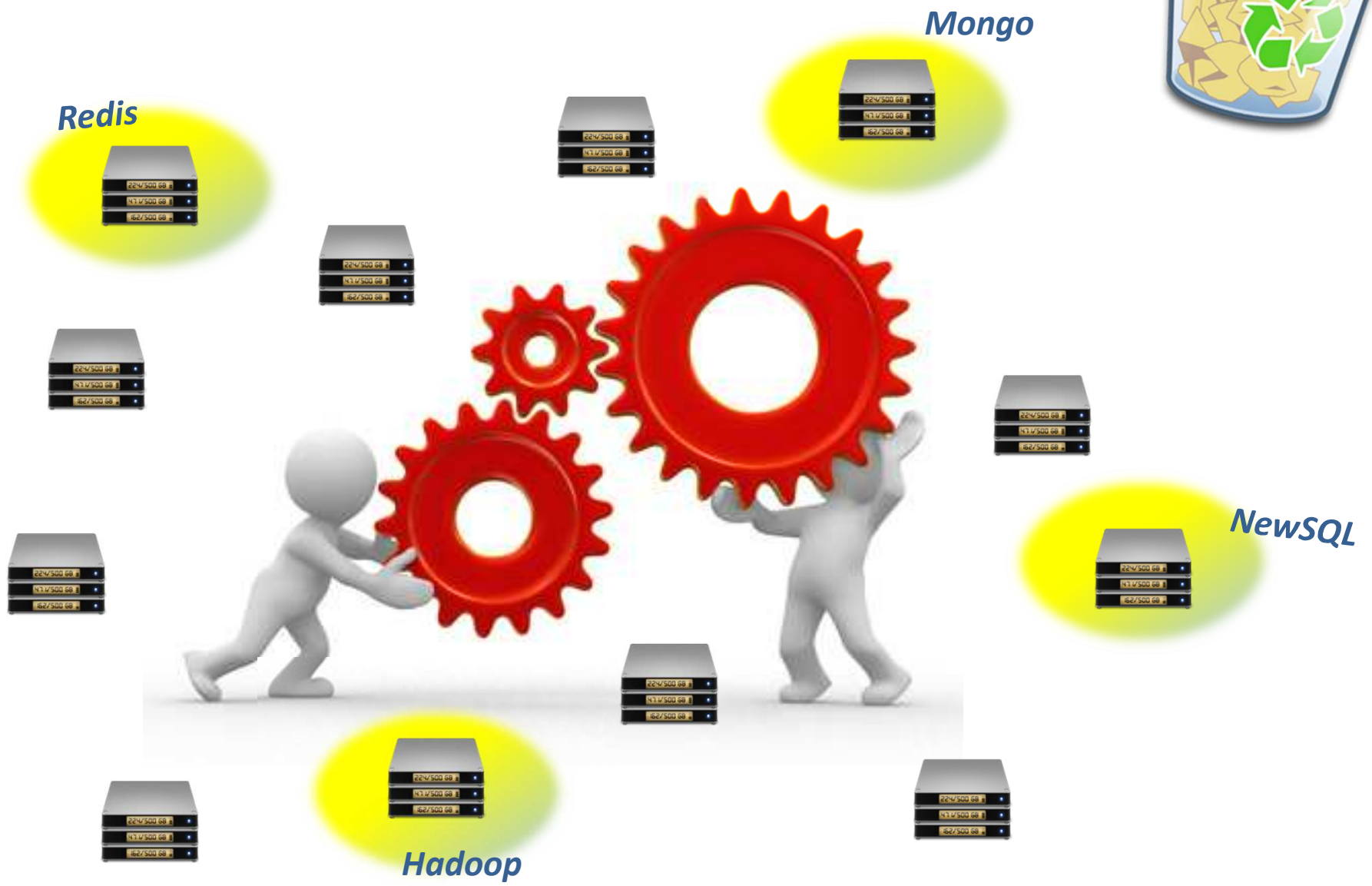
More then one DB is not managable!

One key-player has a favourite DB he knows best!

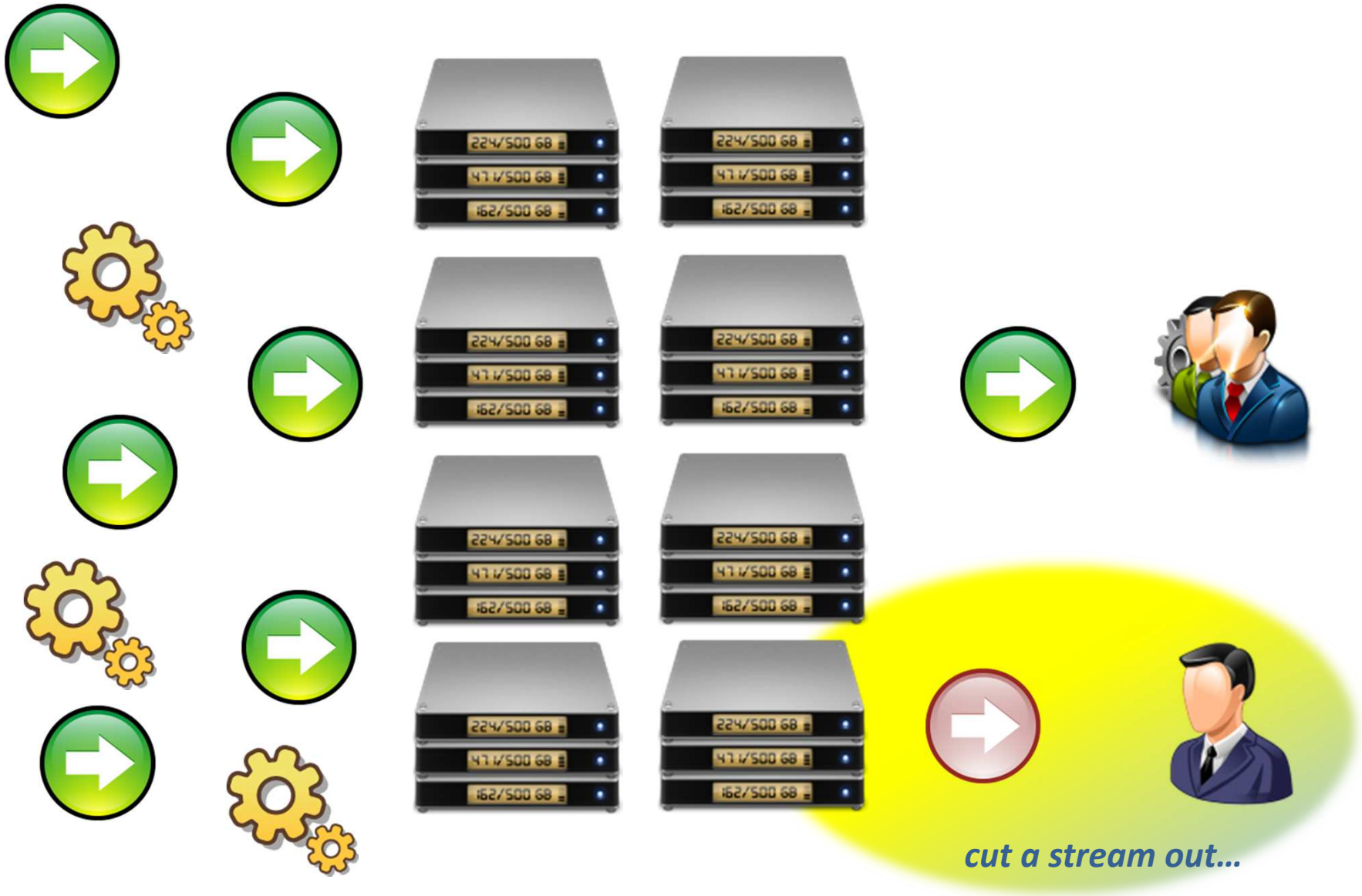
„Give up my personal advantage …“

We do have a 3 year dinosaur contract!

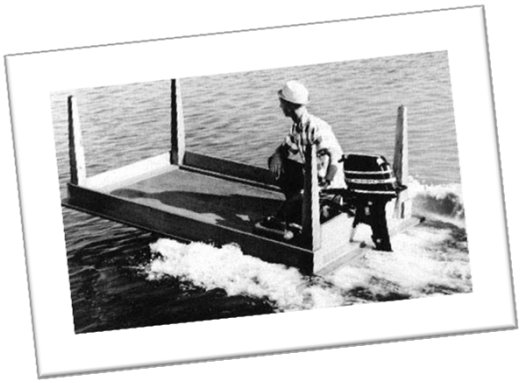
Type A: The db kongo ☺



Type A: The db monolith ☹



Conclusion #1:



Invest a few days in db research & build a prototype is mandatory!

Conclusion #2:

The world is polyglot!

Thanks for your time!

Please contact me:

edlich@gmail.com

edlich.de

copenhagen goto; conference

TALK 2:

NEWSQL! NOSQL UNDER ATTACK

Prof. Dr. Stefan Edlich

Beuth University of Technology Berlin (App.Sc.)

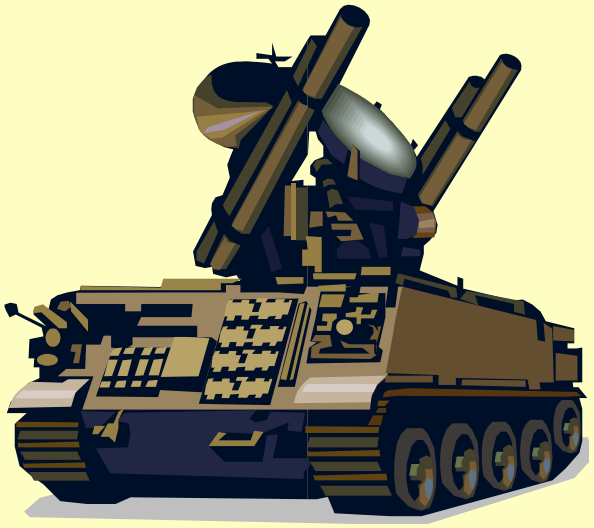
INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

gotocon.com


NoSQL
Ceratosaurus
Eccles Dinosaur Park
Ogden, Utah, USA
icantgetpublished.com

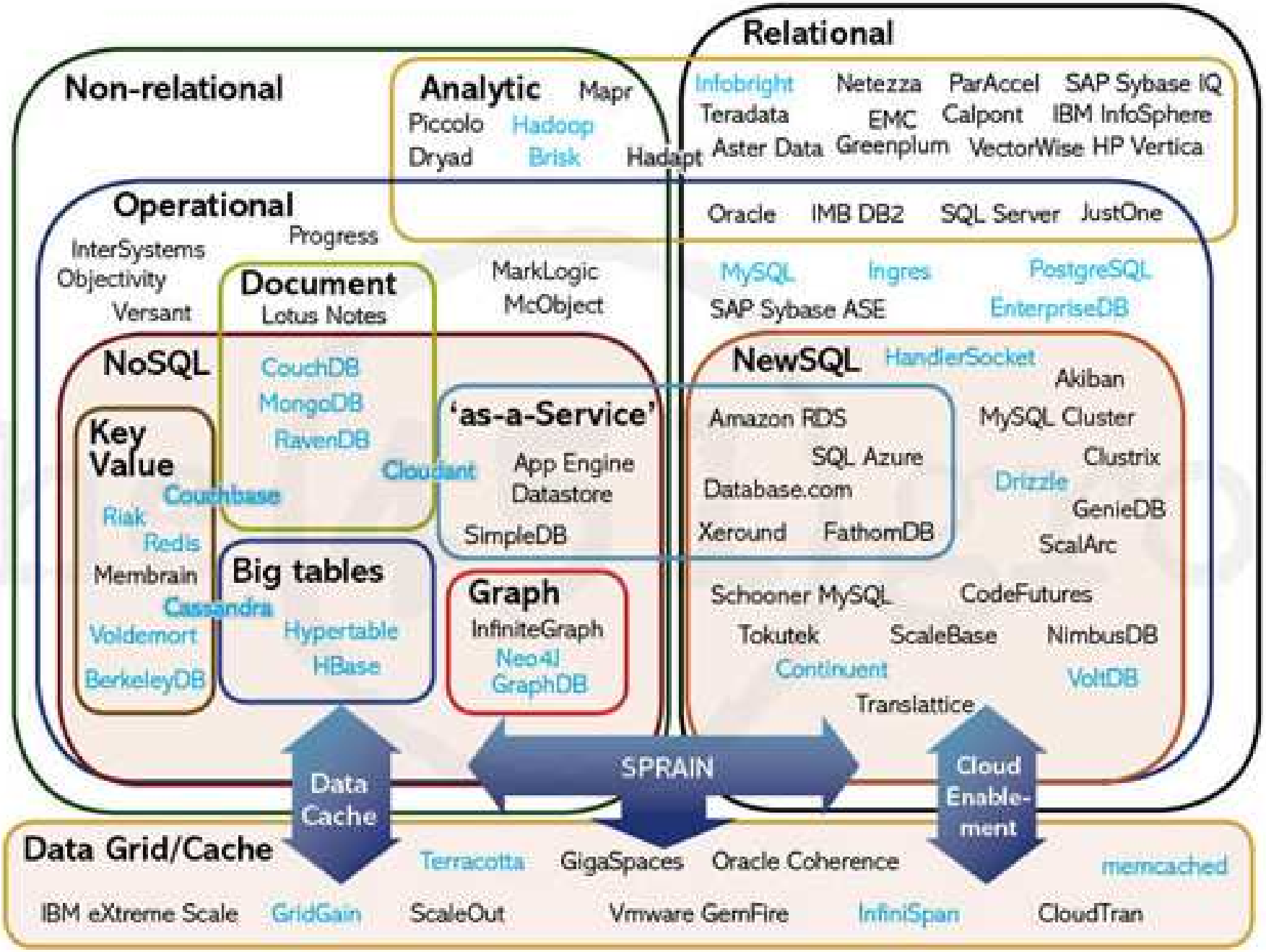
Here are the six urban myths that Mr. Stonebraker says NoSQL advocates incorrectly perpetuate:

- **Myth #1:** SQL is too slow, so use a lower level interface
- **Myth #2:** I like a K-V interface, so SQL is a non-starter
- **Myth #3:** SQL systems don't scale
- **Myth #4:** There are no open source, scalable SQL engines
- **Myth #5:** ACID is too slow, so avoid using it
- **Myth #6:** in CAP, choose AP over CA



MoreSQL

strikes back



Overview

MySQL Extensions
- GenieDB
- ScalArc
- Schooner
- ScaleDB
- Akiban
- Tokutek
- HandlerSocket
- MySQL Cluster

New Engies
- Citrusleaf
- Clustrix
- RethinkDB
- VoltDB
- Translattice
- NimbusDB
- Drizzle
- JustOneDB

Cloud DBs
- Xeround
- FanthomDB
- Database.com
- Amazon RDS
- SQL Azure

GeneralScaler
- CodeFutures / dbShard
- ScaleBase

Latest News

- **MySQL Cluster ++**

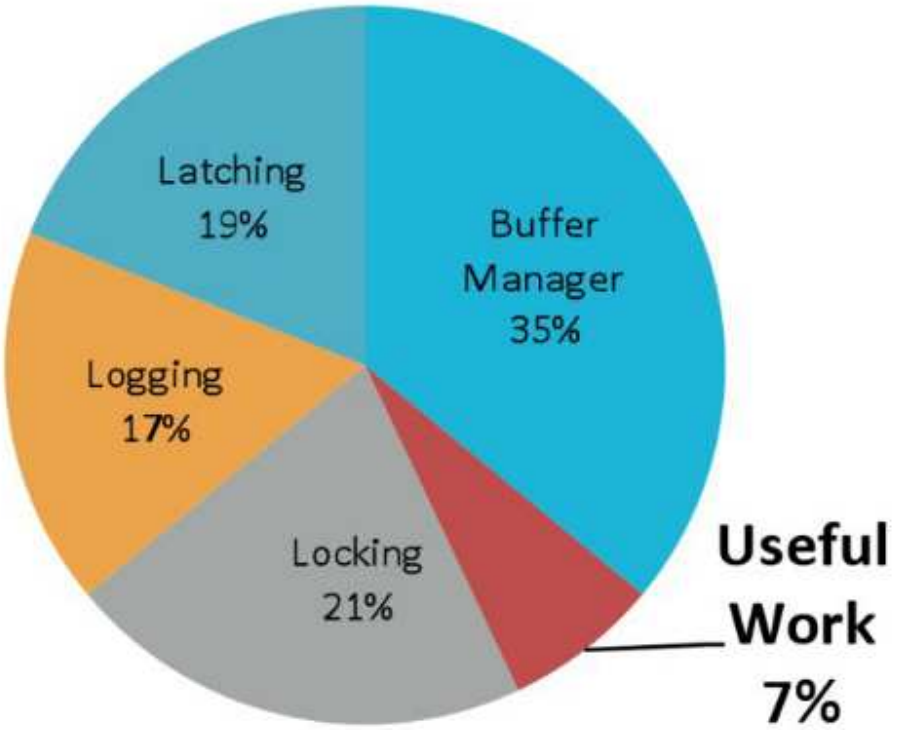
- **PostGres Native JSON + hstore (K/V) support**

- **MySQL 5.6 has a Memcache interface / support: direct K/V access**

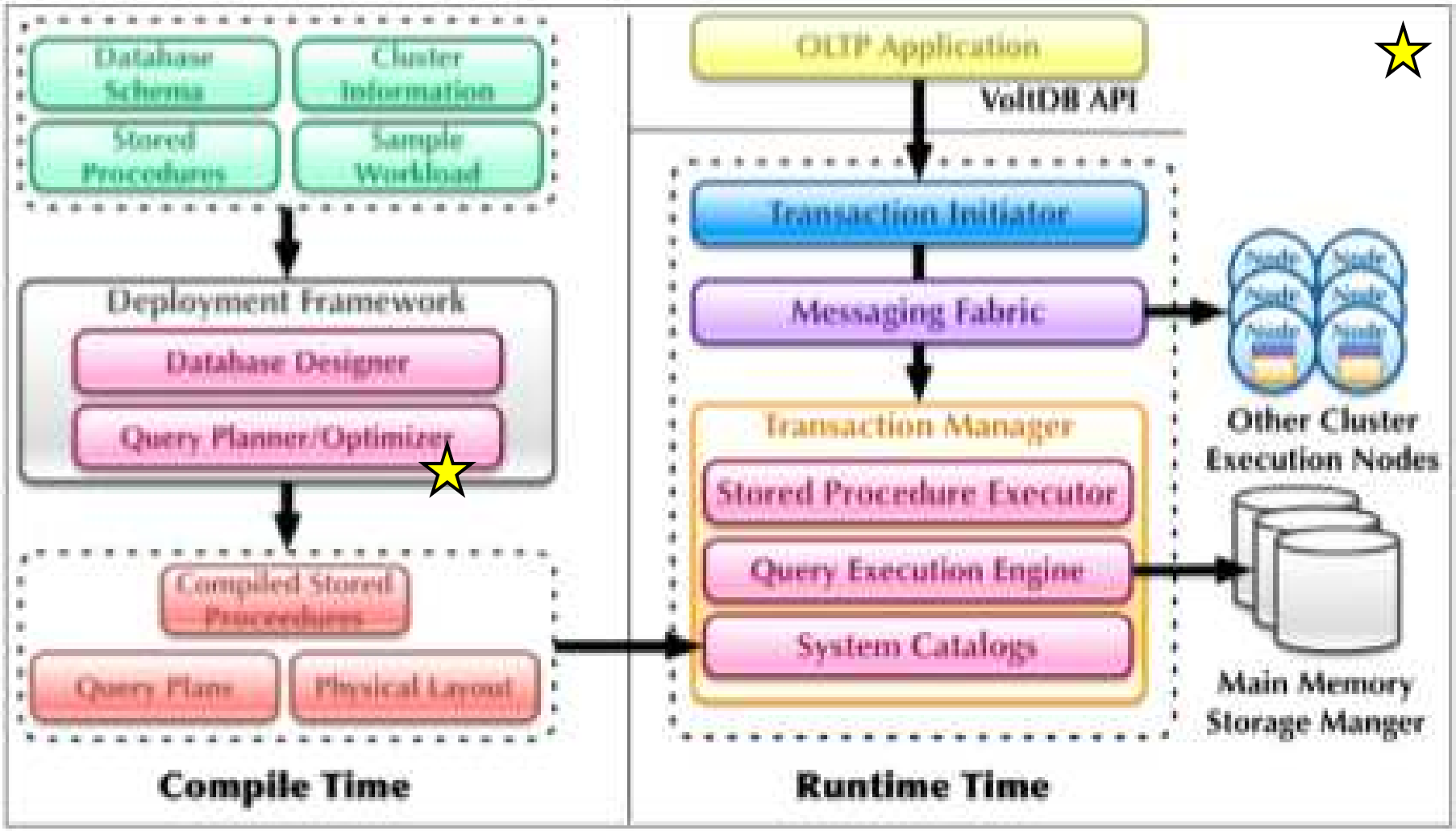
- **Google / Percona / Twitter**
  - **⇨MySQL patches**

VoltDB

Gartner | 2011
COOL VENDOR



Java Stored Procedures!

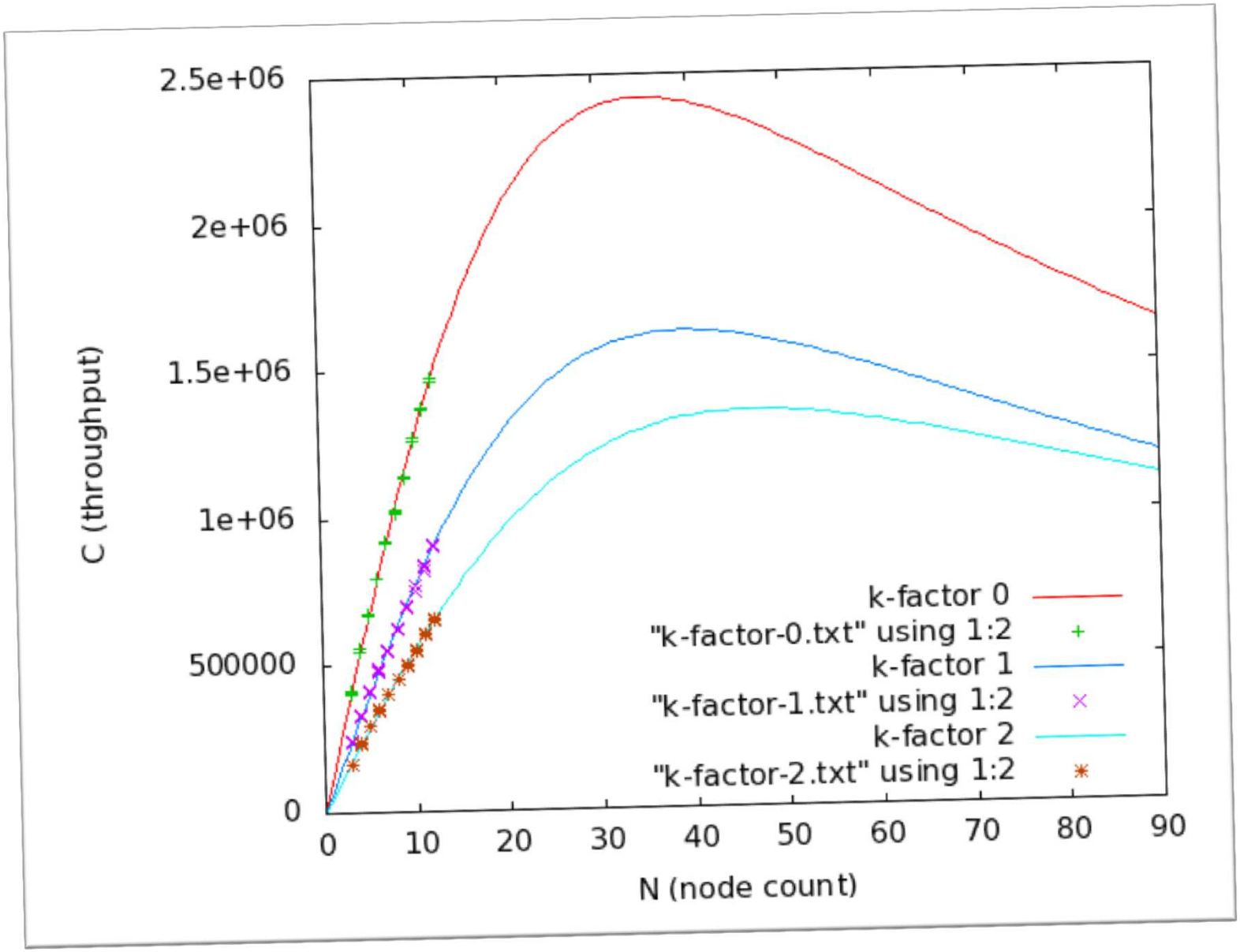


RAM with 100.000 ops/sNode

“VoltDB claims to be 100 times faster than MySQL, up to 13 times faster than Cassandra, and 45 times faster than Oracle, with near-linear scaling.” (highscalability blog)

ACID with partitioned tables

Nearly SQL 99 and ALTER &DROP schema changes require Shutdown static query parametrization



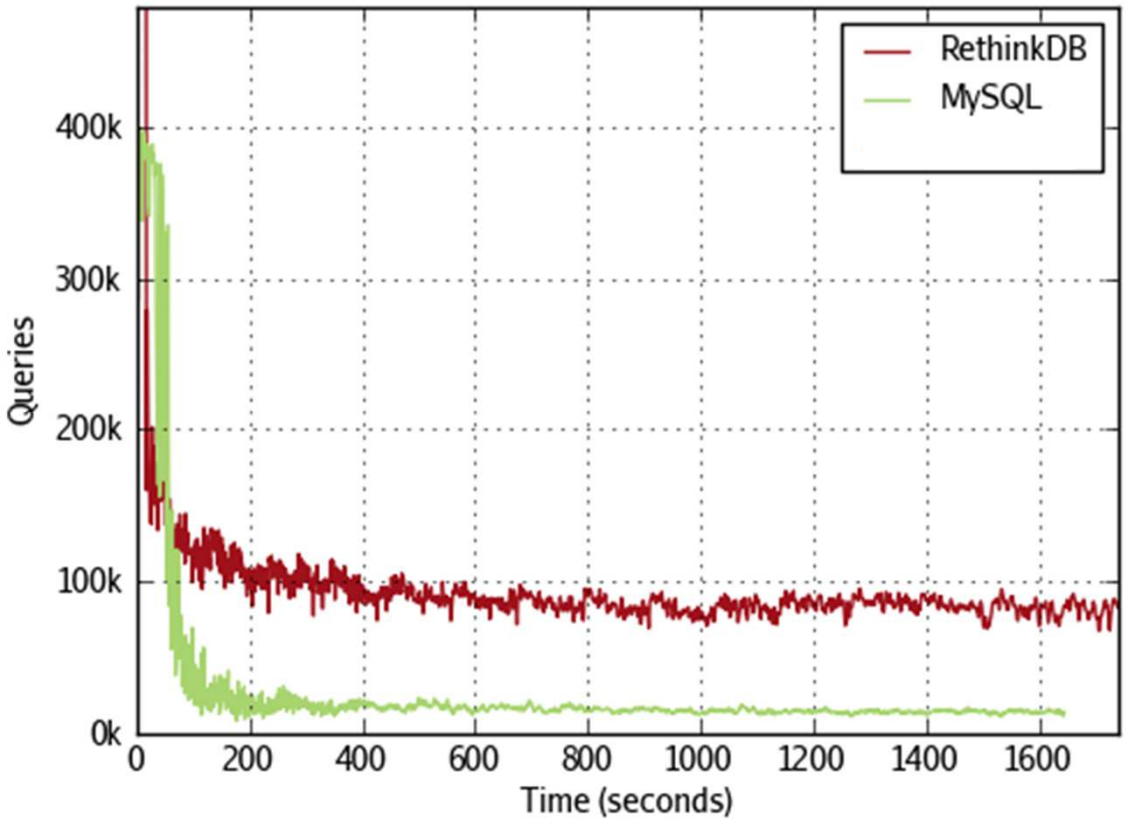
Quelle: Pecond MySQL Performance Blog

NuoDB, GenieDB, ...

RethinkDB.

SSD optimized and disk
C’t: 10-100 TB ok then weaker
10x faster
scaling across cores
random access read pattern

QPS on SSD



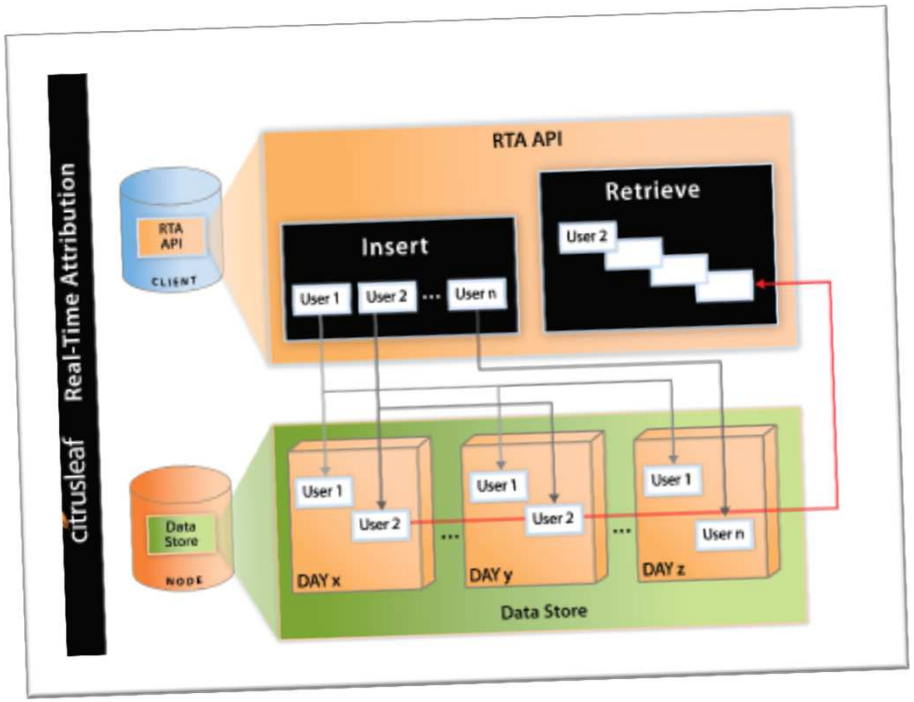
84.426
14.763

5,5 x faster

- **memcached API** more soon
- **no structured data**
- **horizontal scaling for nodes**

citrusleaf™
the real-time noSQL database

- ✓ **"terabytes of data, billions of objects, and 200K plus transactions per second per node, with sub-millisecond latency."**
- ✓ **e.g. real-time bidding**
- ✓ **transactions / ACID**
- ✓ **linear & elastic horizontal scalable**
- ✓ **flash/SSD support**

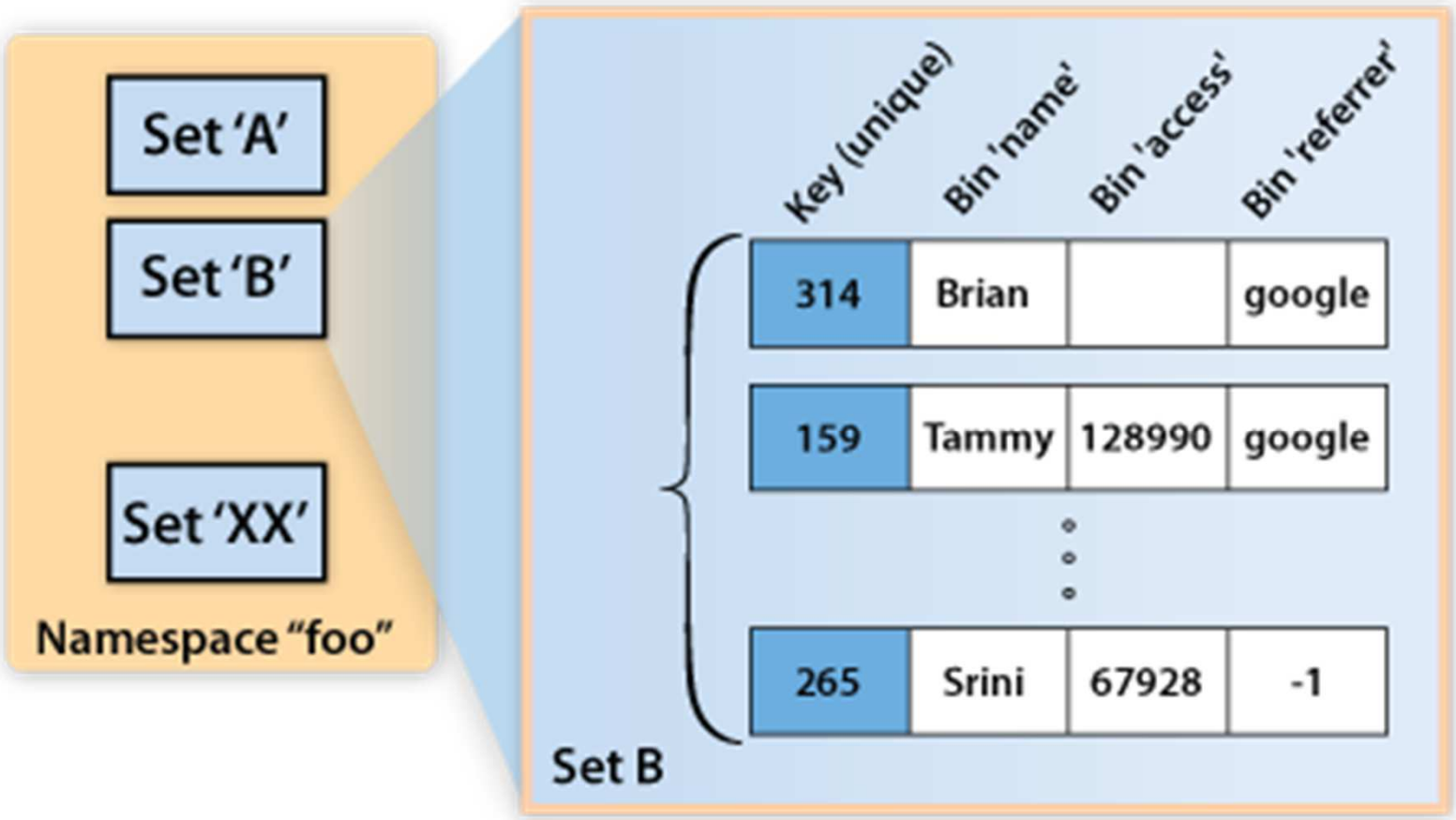


citrusleaf™ RTA™

- ✓ **data expiration**
- ✓ **append list**

citrusleaf™

- ✓ **API: C, C#, Java, Ruby, Python & PHP**
- ✓ **no master node**
- ✓ **200k Ops/secNode read 50k Ops/secNode write**

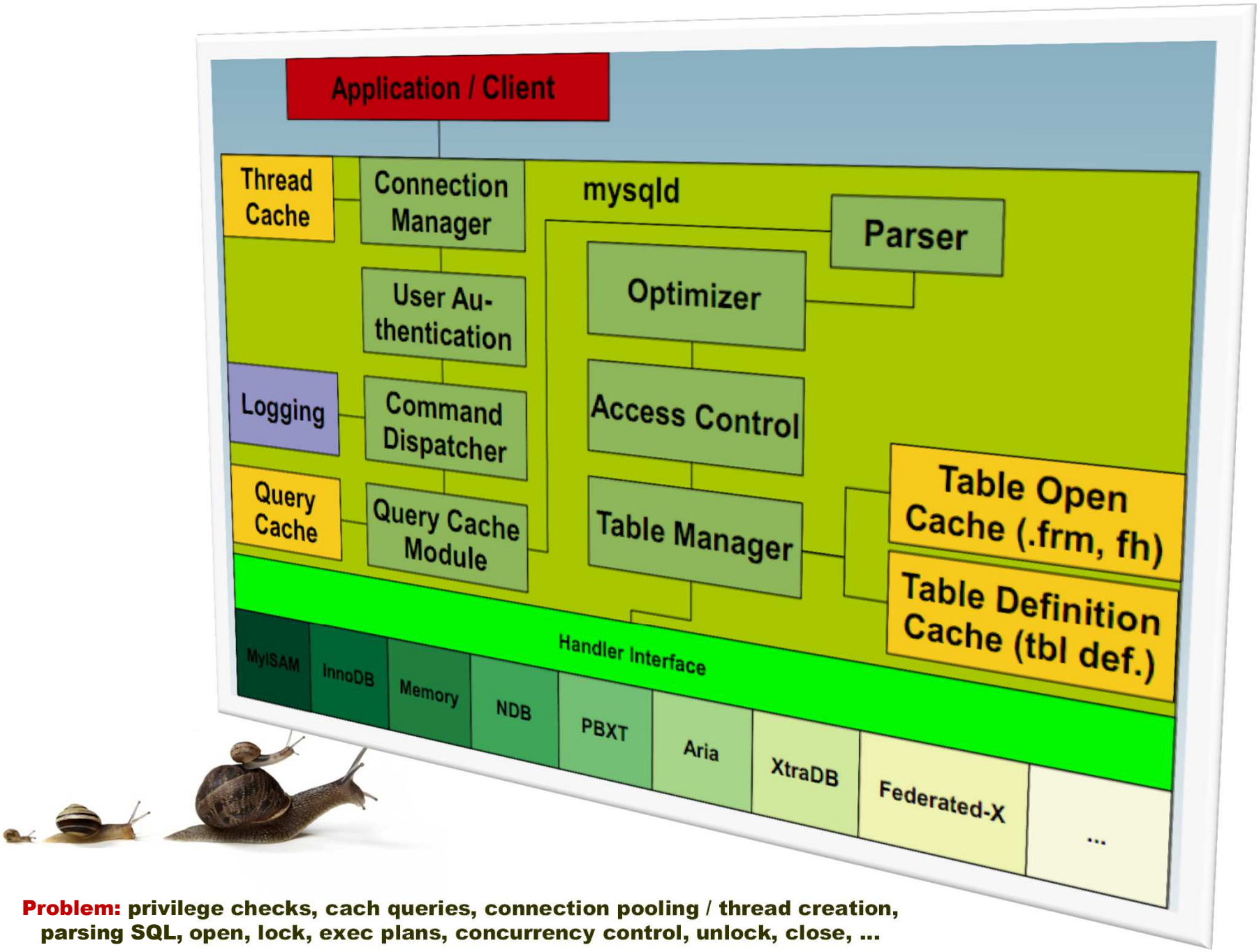


Benchmark	Statistic	Citrusleaf		MongoDB	Redis
#1 Max Throughput • 1 server node • 8-byte ints • 50/50 ratio	Throughput	245,000		31,000	200,000
	Response Time	0.25 ms		0.25 ms	0.15 ms
#2 Max Throughput • 1 server node • 8-byte ints • 95/5 ratio	Throughput	322,000		107,000	232,000
	Response Time	0.24 ms		0.29 ms	0.20 ms
#3 Min Response Time • 1 server node • 8-byte ints • 50/50 ratio	Throughput	50,000		N/A[2]	50,000
	Response Time	0.14 ms		N/A[2]	0.11 ms
#4 Min Response Time • 1 server node • 8-byte ints • 95/5 ratio	Throughput	50,000		50,000	50,000
	Response Time	0.13 ms		0.18 ms	0.11 ms
#5 Memory Efficiency • 1 server node • 8-byte ints	Memory Used Per 1M Objects	67 MB		221 MB	163 MB
#6 Replication: Max Throughput • 2 server nodes, with Replication • 8-byte ints • 50/50 ratio	Immediate Consistency[1]	**Yes**		**Yes**[4]	*No*
	Client-Side Throughput[3]	200,000		26,000	166,000
	Server-Side Throughput[3]	299,000		40,000	166,000
	Response Time	0.55 ms		1.20 ms	0.18 ms
#7 Replication: Max Throughput • 2 server nodes, with Replication • 8-byte ints • 95/5 ratio	Immediate Consistency	**Yes**		**Yes**[4]	*No*
	Client-Side Throughput[3]	526,000		96,000	221,000
	Server-Side Throughput[3]	552,000		101,000	221,000
	Response Time	0.36 ms		0.33 ms	0.21 ms
#8 Large Objects • 1 server node • 1,024 char Strings • 50/50 ratio	Objects in Memory	**No**[5]	*Yes*[5]	*Yes*	*Yes*
	Throughput	71,000	163,000	25,000	138,000
	Response Time	0.70 ms	0.49 ms	0.40 ms	0.48 ms
	Memory Used Per 1M Objects	70 MB	1,077 MB	1,209 MB	1,177 MB

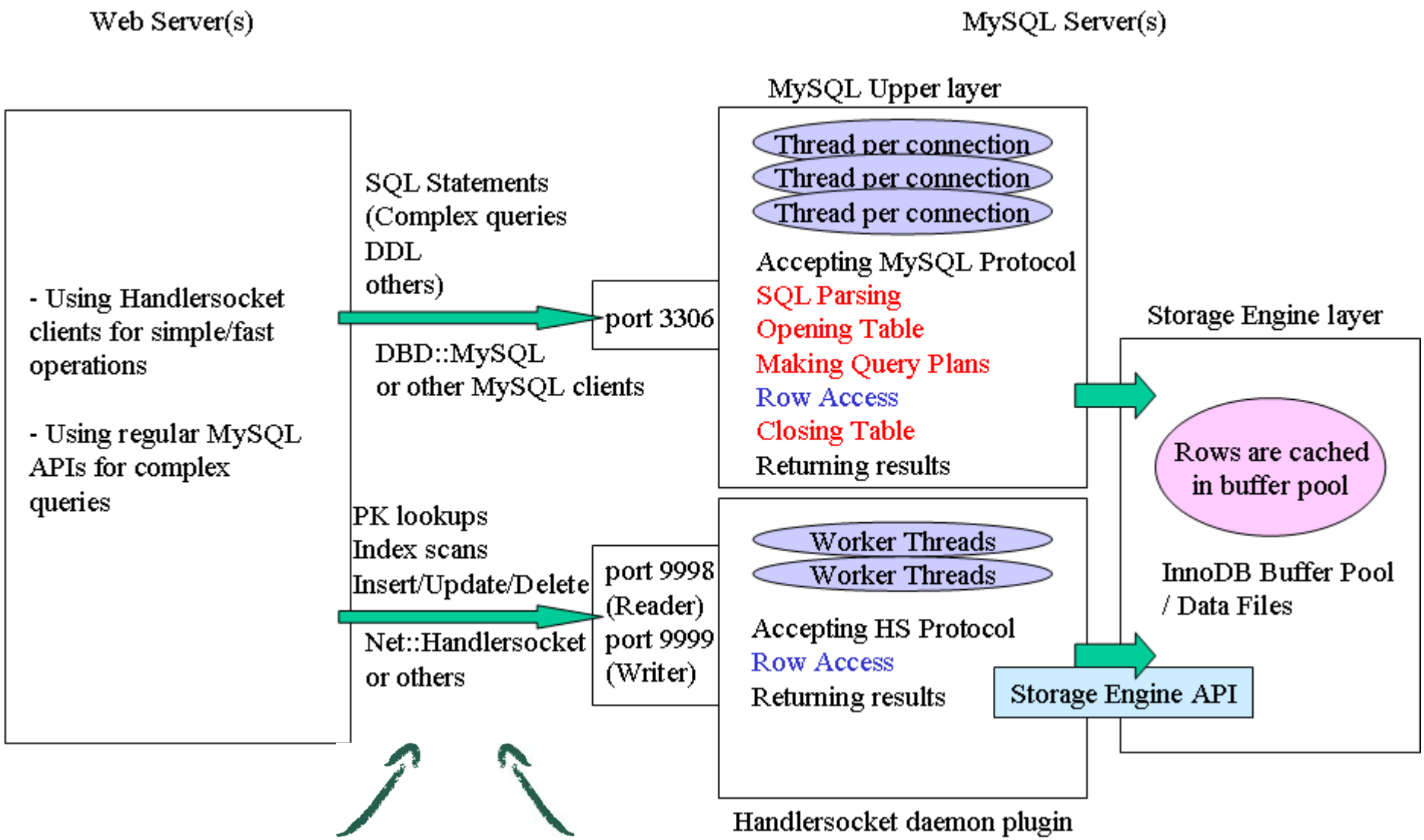
Benchmark	Server Nodes	Throughput (Server-Side[2])	Response Time (Reads)	Response Time (Updates)
#9 Scalable Throughput • 2, 3 and 4 nodes, with Replication • Constant ratio of clients to hosts • 8-byte ints • 50/50 ratio	2	297,000	0.26 ms	0.69 ms
	3	404,000	0.28 ms	0.77 ms
	4	519,000	0.29 ms	0.80 ms
#10 Scalable Throughput [6] • 2, 3 and 4 nodes, with Replication • Constant ratio of clients to hosts • 8-byte ints • 95/5 ratio	2	433,000	0.30 ms	0.39 ms
	3	636,000	0.30 ms	0.41 ms
	4	839,000	0.31 ms	0.42 ms
#11 Scalability: Min Response Time • 2, 3 and 4 nodes, with Replication • 8-byte ints • 50/50 ratio	2	100,000	0.16 ms	0.34 ms
	3	150,000	0.16 ms	0.33 ms
	4	200,000	0.16 ms	0.34 ms

Check hybrid solutions!

easier & better then RDBMS + memcache



Problem: privilege checks, cach queries, connection pooling / thread creation, parsing SQL, open, lock, exec plans, concurrency control, unlock, close, ...



keep tables open & simple protocol

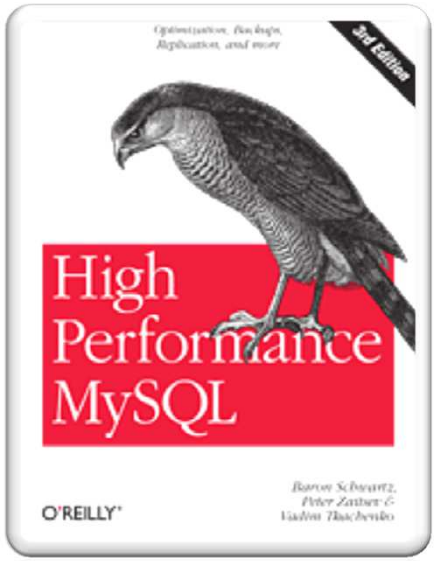
QUELLE: YOSHINORI MATSUNOBU

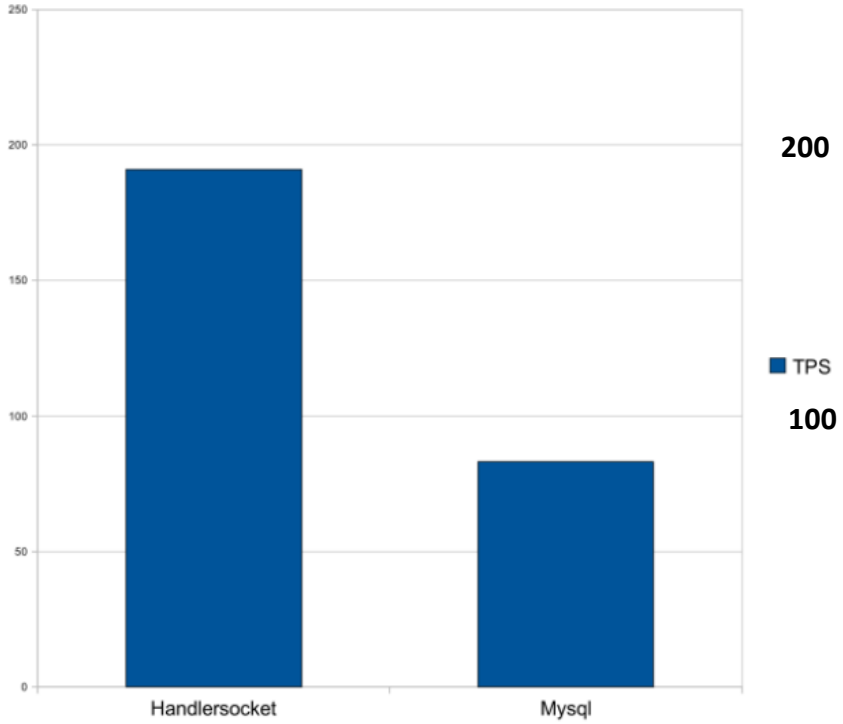
Performance ✓
Transactions ✓
Concurrent Access ✓
No Cache / Crash-Safe ✓
no SQL but more then ✓

K/V: ranges, LIMIT, multi_get,...

no Security
new API

	approx qps	server CPU util
MySQL via SQL	105,000	%us 60% %sy 28%
memcached	420,000	%us 8% %sy 88%
MySQL via HandlerSocket	750,000	%us 45% %sy 53%





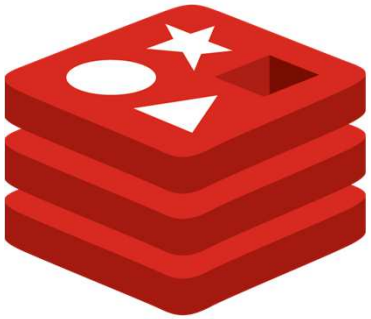
PERCONA

Conclusion #1

Conclusion #2

There is no
“one perfect solution”

Check hybrid solutions
and NewSQL DBs too!



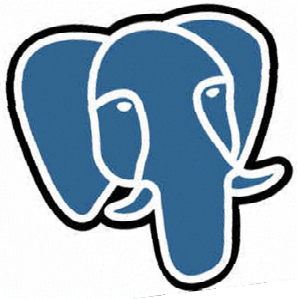
redis

- **main feed**
- **activity feed**
- **sessions**

Must have:

- **Quick K/V lookup => hashes**
- **Fit in Memory (17-34)**
- **Be persistent**

PostgreSQL

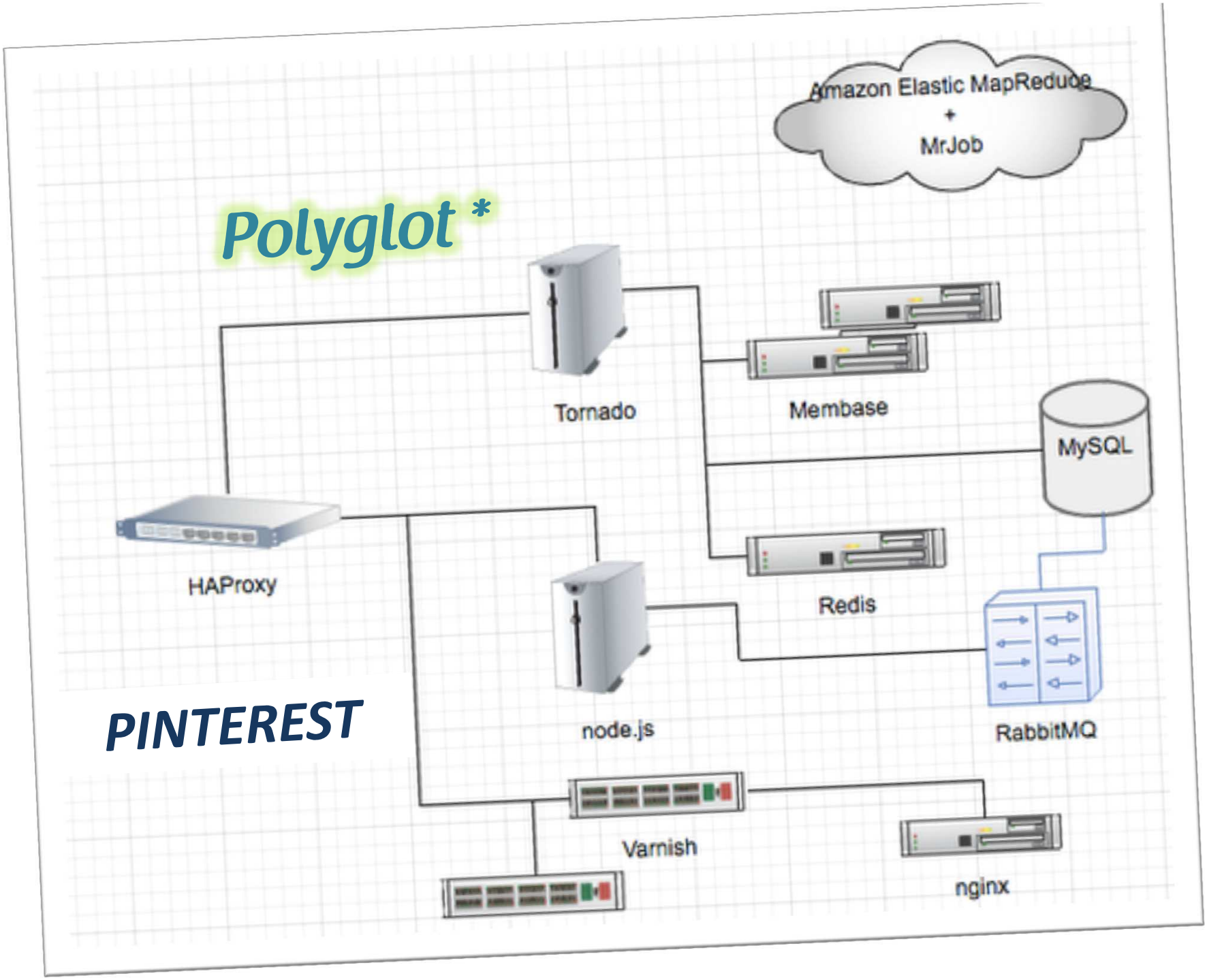


12 sharded quadruple extra-large Instances

S3

One simple solution to this problem would be to simply store them as a bunch of rows in a database, with "Media ID" and "User ID" columns. However, a SQL database seemed like overkill given that these IDs were never updated (only inserted), didn't need to be transactional, and didn't have any relations with other tables.

Instead, we turned to Redis, an advanced key-value store that we use extensively here at Instagram (for example, it powers our main feed). Redis is a key-value swiss-army knife; rather than just normal "Set key, get key" mechanics like Memcached, it provides powerful aggregate types like sorted sets and lists. It has a configurable persistence model, where it background saves at a specified interval, and can be run in a master-slave setup. All of our Redis deployments run in master-slave, with the slave set to save to disk about every minute.


Amazon Elastic MapReduce
+
MrJob
Polyglot *
Tornado
Membase
MySQL
HAProxy
Redis
node.js
RabbitMQ
PINTEREST
Varnish
nginx

Thanks!

Contact me please!

edlich@gmail.com
edlich.de