EVALUATING POLYGLOT PERSISTENCE

BUT I NEED A DATABASE THAT SCALES!

MICROSOFT

MYSQL

POSTGRESQL

ORACLE

MICROSOFT

MYSQL

POSTGRESQL

ORACLE

MONGODB

RIAK

GREENPLUM

COUCHDB

NEO4J

CASSANDRA

HDFS

REDIS

The Register®
Biting the hand that feeds IT

Internet-of-stuff startup dumps NoSQL for ... SQL?

NoSQL taste
cloud whiz

http://techcrunch.com/2010/09/07/digg-struggles-vp-engineering-door/

As Digg Struggles, VP Of Engineering Is Shown The Door

Posted Sep 7, 2010 by Erick Schonfeld (@erickscho

-3
SHARES

Ever since Digg launched its new site
design, it's been plagued with all kin
trouble, not least of which is that it
going down. Th
architecture are
Engineering Joh

NoSQL Mee
Down Two
Flexcoin an

confirmed with

In a Diggnation
Rose explained

Emin Gün Sirer

← Older

Newer →

Flexcoin was a Bitcoin exchange that shut down on March 3rd, 2014, when someone allegedly hacked in and made off with 896 BTC in the hot wallet. Because the half-million dollar heist from the hot wallet was too large for the company to bear, it folded.

I'll resist the urge to ask why they did not have deposit insurance for

http://hackingdistributed.com/2014/04/06/another-one-bites-the-dust-flexcoin/

Database Horror
s: healthcare.gov

Share
+

e still awake. At wit's
system is
as you pick up your
)."

ries-healthcare-gov

Tuesday, 23 February 2010

are it:

med rumors about Twitter planning to use Cassandra for a long time.
ed post, I couldn't find any other references.

and we all know that NoSQL projects love Twitter. So, imagine how
er posting about Cassandra 0.5.0 release, I received a short email from
Cassandra efforts at Twitter simply saying that he would be glad to talk

, here is the conversation I had with Ryan King (@rk) about **Cassandra**

http://nosql.mypopescu.com/post/407159447/cassandra-twitter-an-interview-with-ryan-king

Sarah Mei

« What Your Conference Proposal Is Missing

How To Prevent Inappropriate Presentations »

Why You Should Never Use MongoDB

Disclaimer: I do not build database engines. I build web applications. I run 4-6 different projects every year, so I build **a lot** of web applications. I see apps with different requirements and different data storage needs. I've deployed most of the data stores you've heard about, and a few that you probably haven't.

I've picked the wrong one a few times. This is a story about one of those times — why we picked it originally, how we discovered it was wrong, and how we recovered. It all happened on an open source project called Diaspora.

The project

Diaspora is a distributed social network with a long history. Waaaaay back in early 2010, four

http://www.sarahmei.com/blog/2013/11/11/why-you-should-never-use-mongodb/

Why I Migrated Away From

I recently concluded a migration away from
digiDoc. I'd like to tell you why I did so.

To be honest, the decision to use MongoDb
thoroughly research any new technology yo
and weaknesses thereof and evaluate hones
much hype there is surrounding said techn
compaints against MongoDB such as data
and so on. For our use case, mongodb faile

digiDoc is all about converting paper docur

http://svs.io/post/31724990463/why-i-migrated-away-from-

AARON'S THESIS

- Developers aren't (entirely) to blame.

- Database websites and communities don't offer enough data about trade-offs and drawbacks.

- What we need is a **systematic approach** to evaluating databases.

AGENDA

- What **database resources** are we scaling?
- What **RDBMS characteristics** are up for grabs with new architectures?
- Take a look at common **scaling practices** to evaluate trade offs.
- Some sample **database evaluations** .

WHAT EVEN IS A DATABASE?

da·ta·base

/ˈdatəˌbās,ˈdā-/

noun

noun: **database**; plural noun: **databases**

a structured set of data held in a computer, especially one that is accessible in various ways.

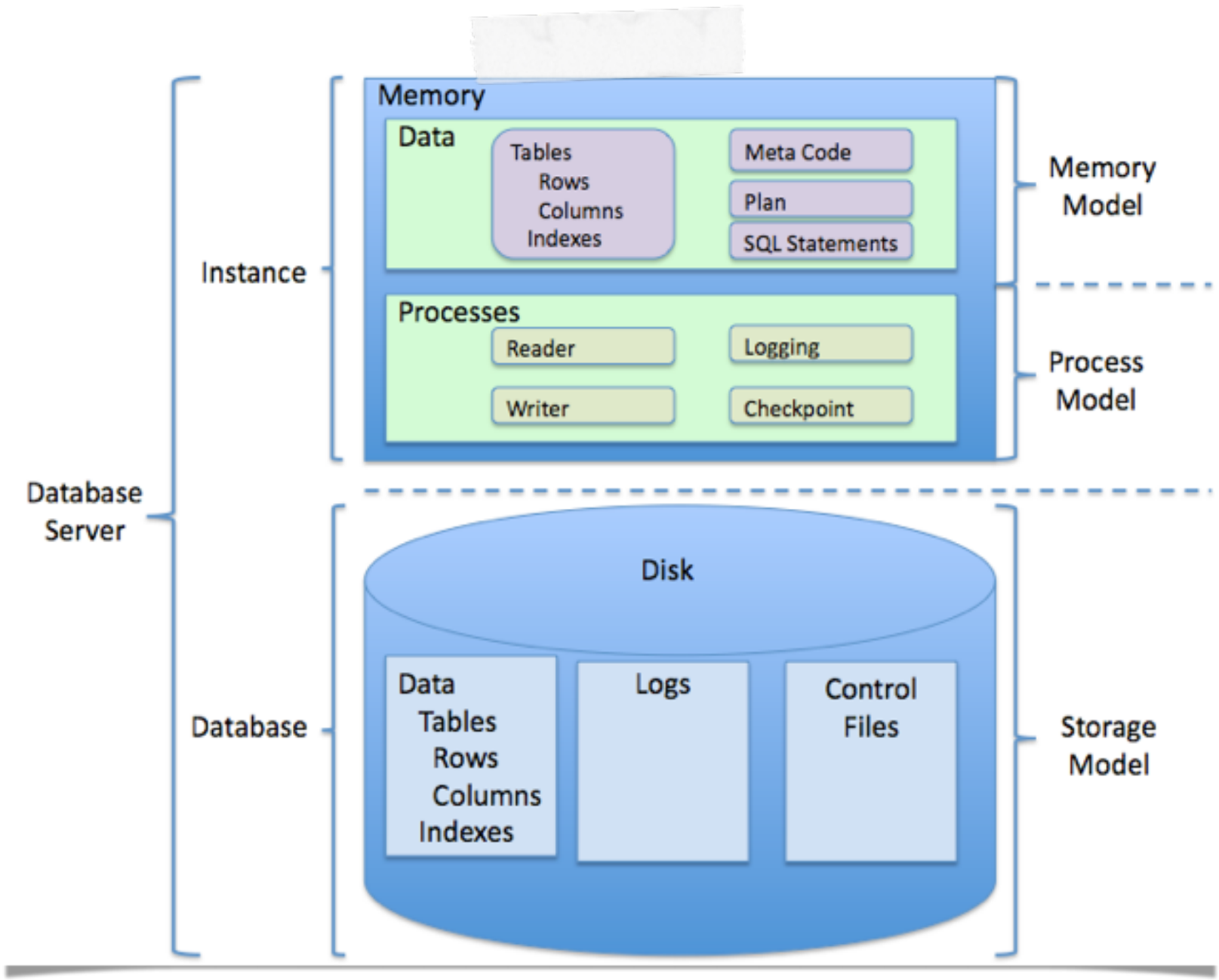
Translate database to Choose language

Use over time for: database

Show less

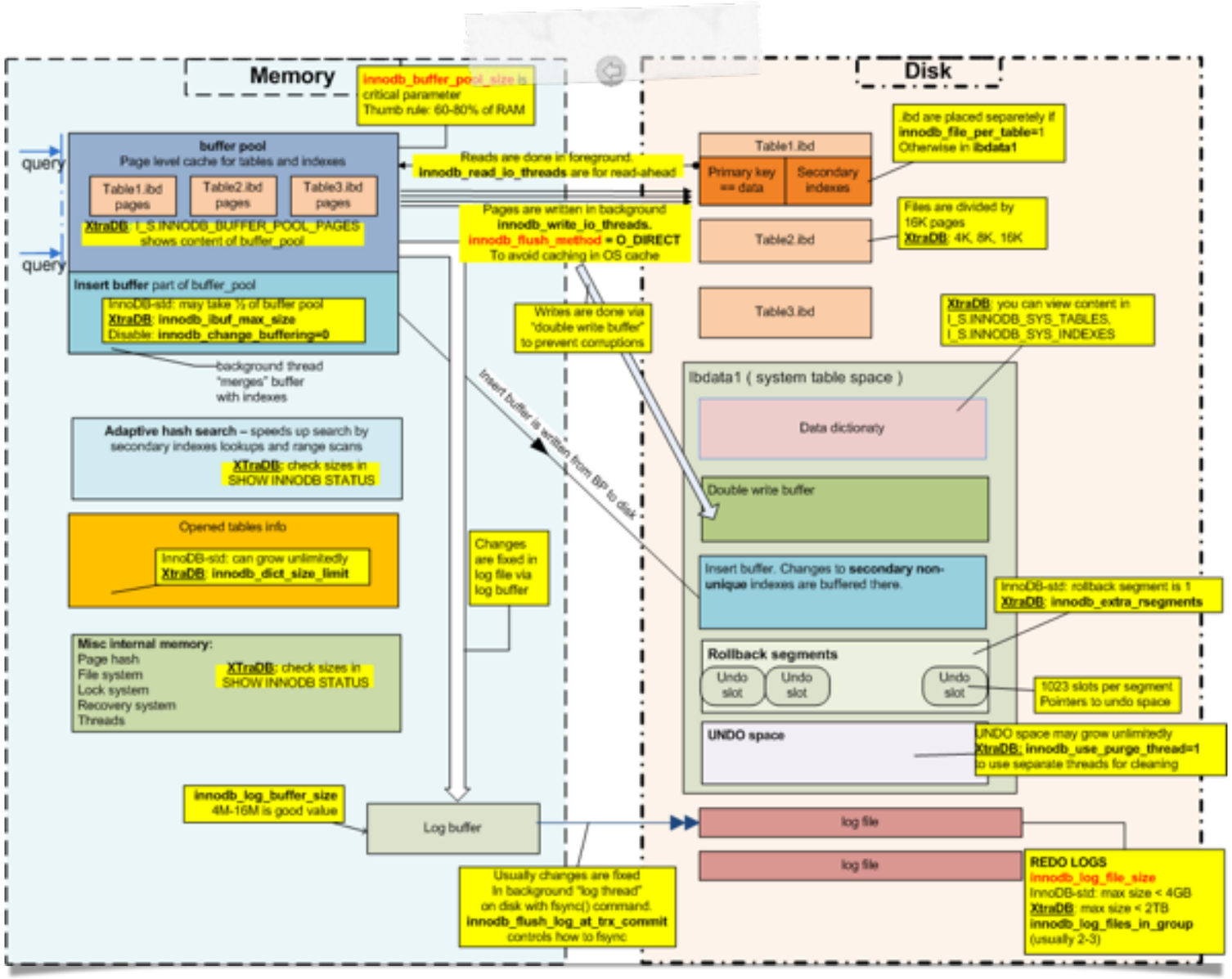
Google Search Definition

WHAT EVEN IS A DATABASE?



https://en.wikipedia.org/wiki/Relational_database_management_system#/media/File:RDBMS_structure.png
By Scifipete - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=11506013

WHAT EVEN IS A DATABASE?

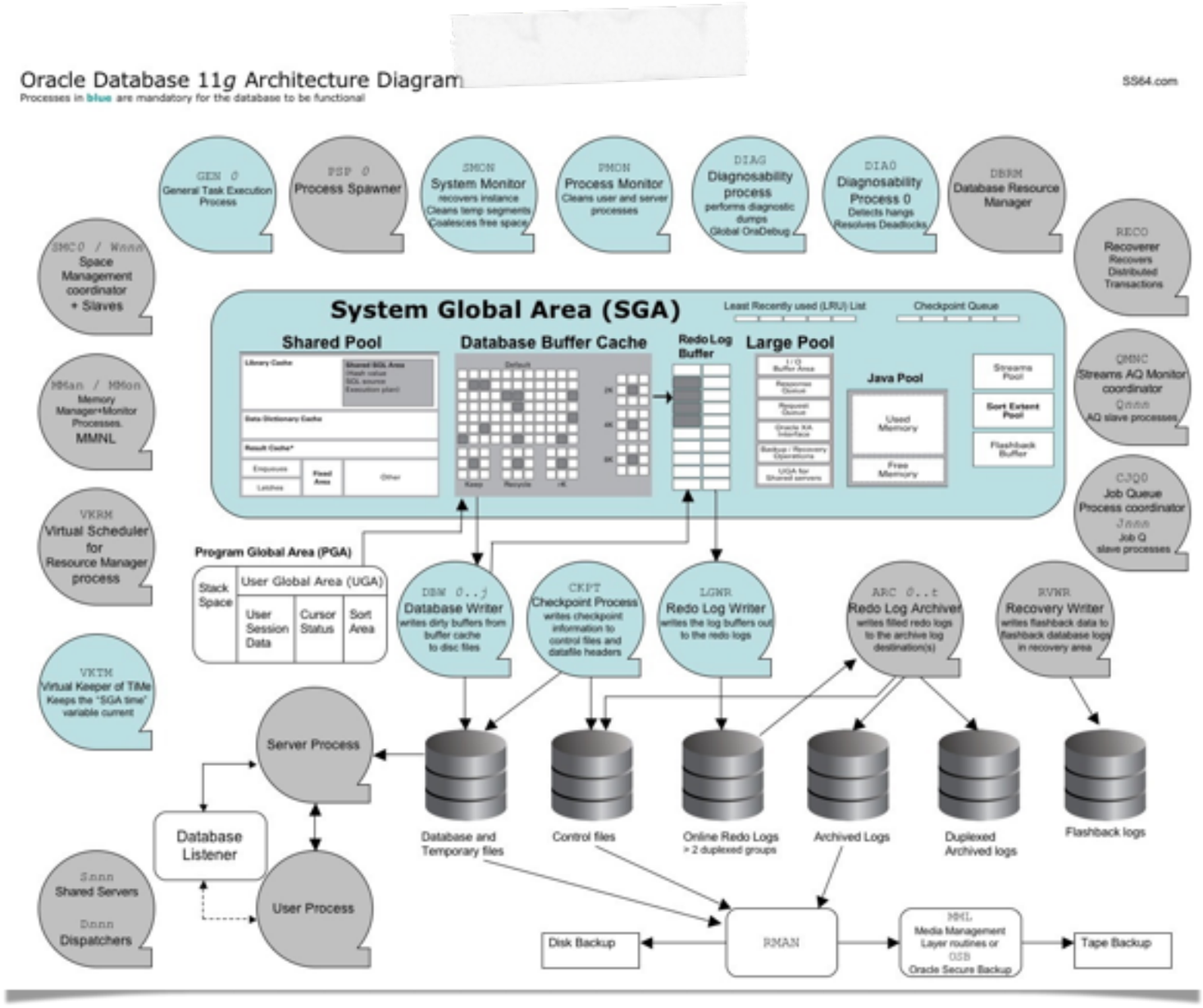


MySQL InnoDB

Vadim Tkachenko

https://www.percona.com/blog/2010/04/26/xtradb-innodb-internals-in-drawing/

WHAT EVEN IS A DATABASE?



http://ss64.com/ora/syntax-architecture.html

WHAT EVEN IS A DATABASE?

- A **collection** of processes that collaborate to store, retain & query data.
- Sometimes a **distributed system**.

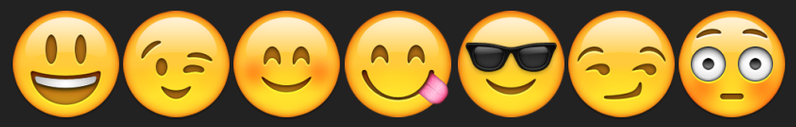
WHEN YOU ARE STUCK IN A TRAFFIC JAM WITH A PORSCHE, ALL YOU DO IS BURN MORE GAS IN IDLE.

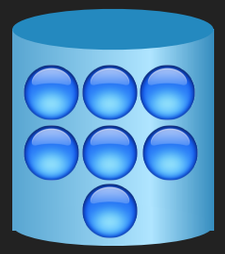
SCALABILITY IS ABOUT BUILDING WIDER ROADS, NOT ABOUT BUILDING FASTER CARS.

Steve Swartz

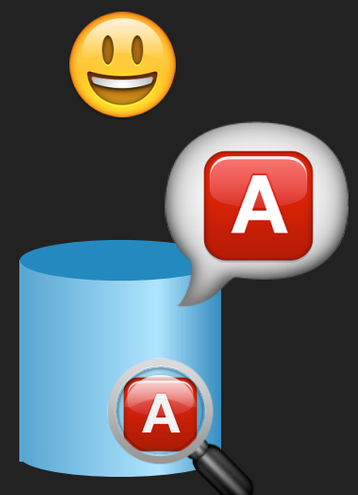
RESOURCES: WHAT DO WE MEAN BY "SCALE"

Connections





Read Transactions

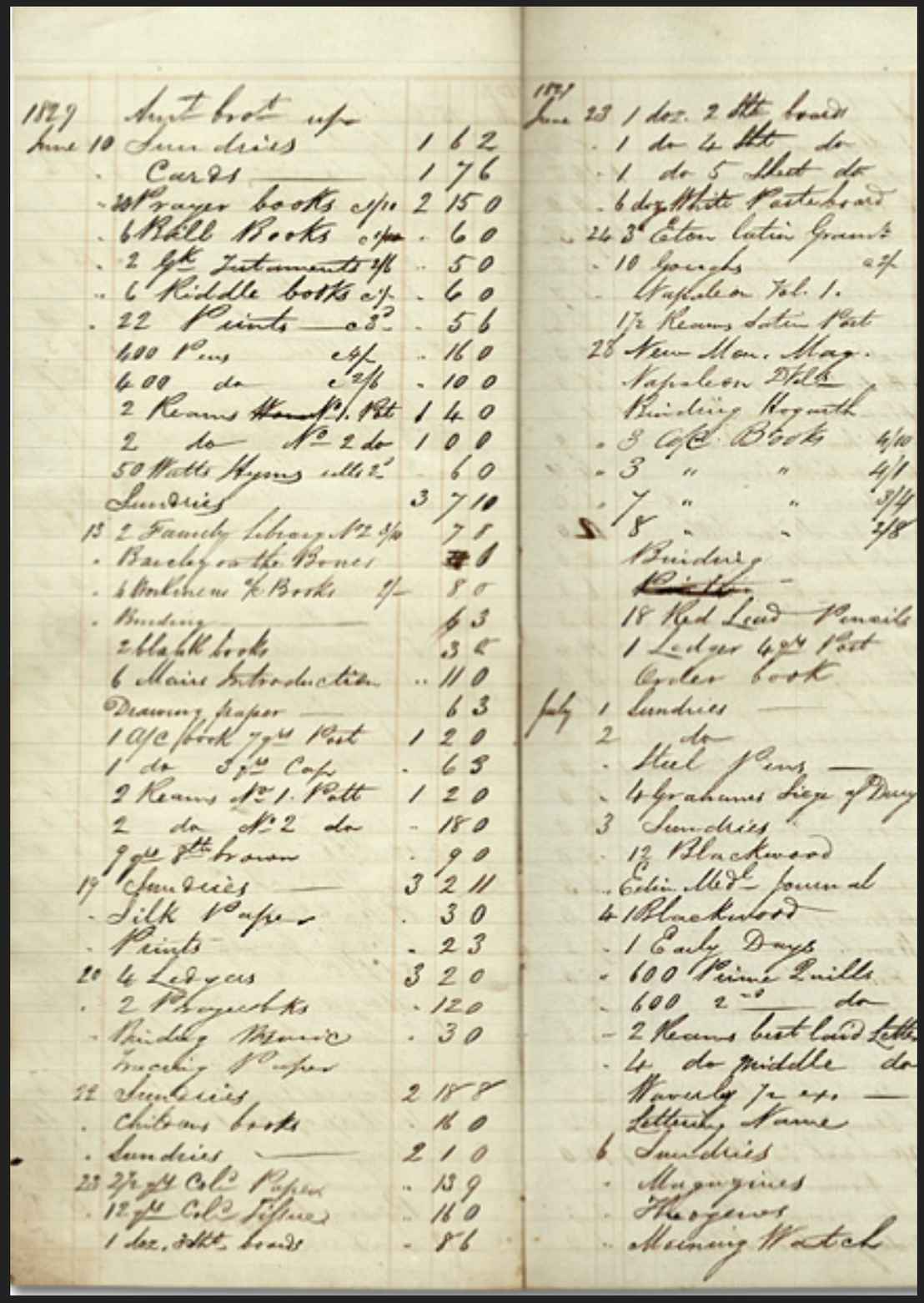


Write Transactions

Scanning/Analysis

CHARACTERISTICS OF THE RDBMS

- [Mostly] Ledger or Table Orientation
- Relational Operations: Groupings, Joins, Conditional Clauses, etc.
- **ACID** Characteristics



A • C • I • D

EXPLAINED IN EMOJI 😀

Atomicity	Consistency	Isolation	Durability
			

INNOVATION [IN COMPUTER SCIENCE] ISN'T ABOUT CREATING SOMETHING NEW.

IT'S ABOUT MAKING DIFFERENT TRADEOFFS.

Unattributed

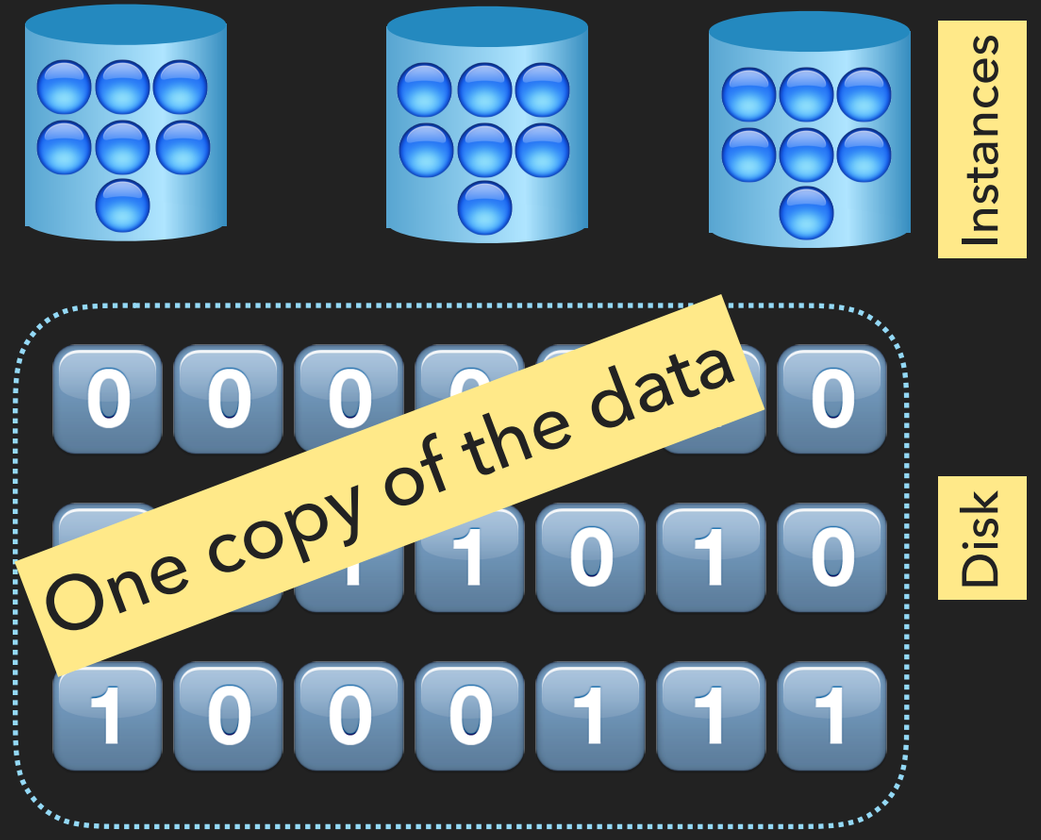
COMMON SCALING PRACTICES

- Connection Scaling
- Read Caching
- Write Coalescence
- Vertical Partitioning
- Horizontal Partitioning

SCALING PRACTICES

Connection Scaling

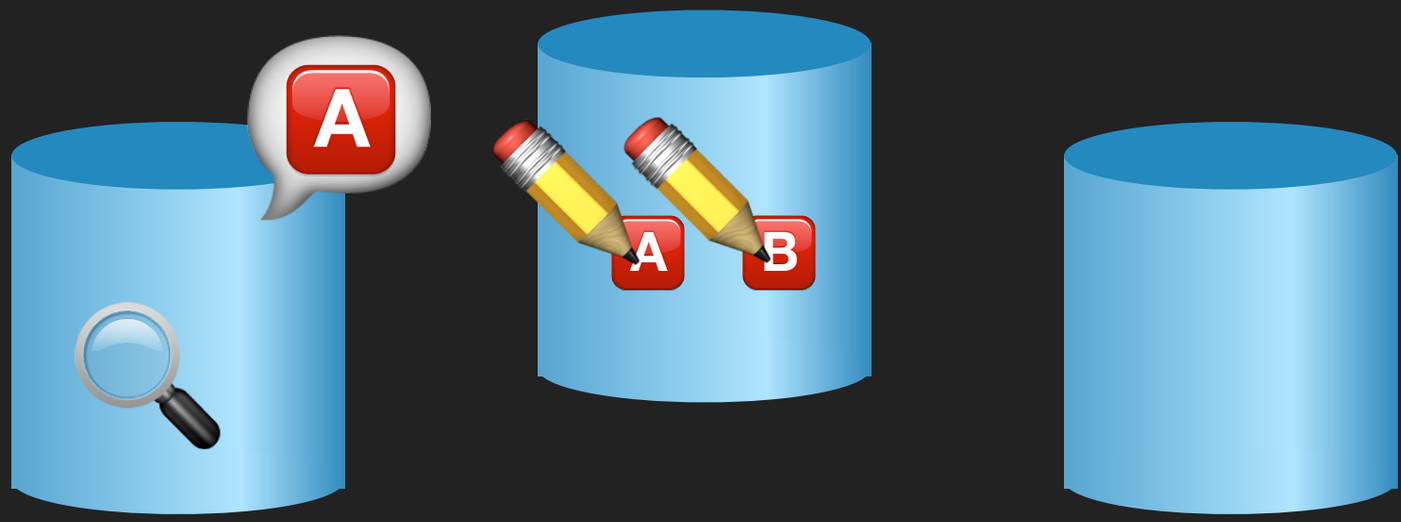
- Increases connection count & accessible RAM pool for internal caching



- Cache misses, high throughput writes & full-table scans remain unimproved
- Generally, no negative impact on relational characteristics

SCALING PRACTICES

Master/Agent Replication

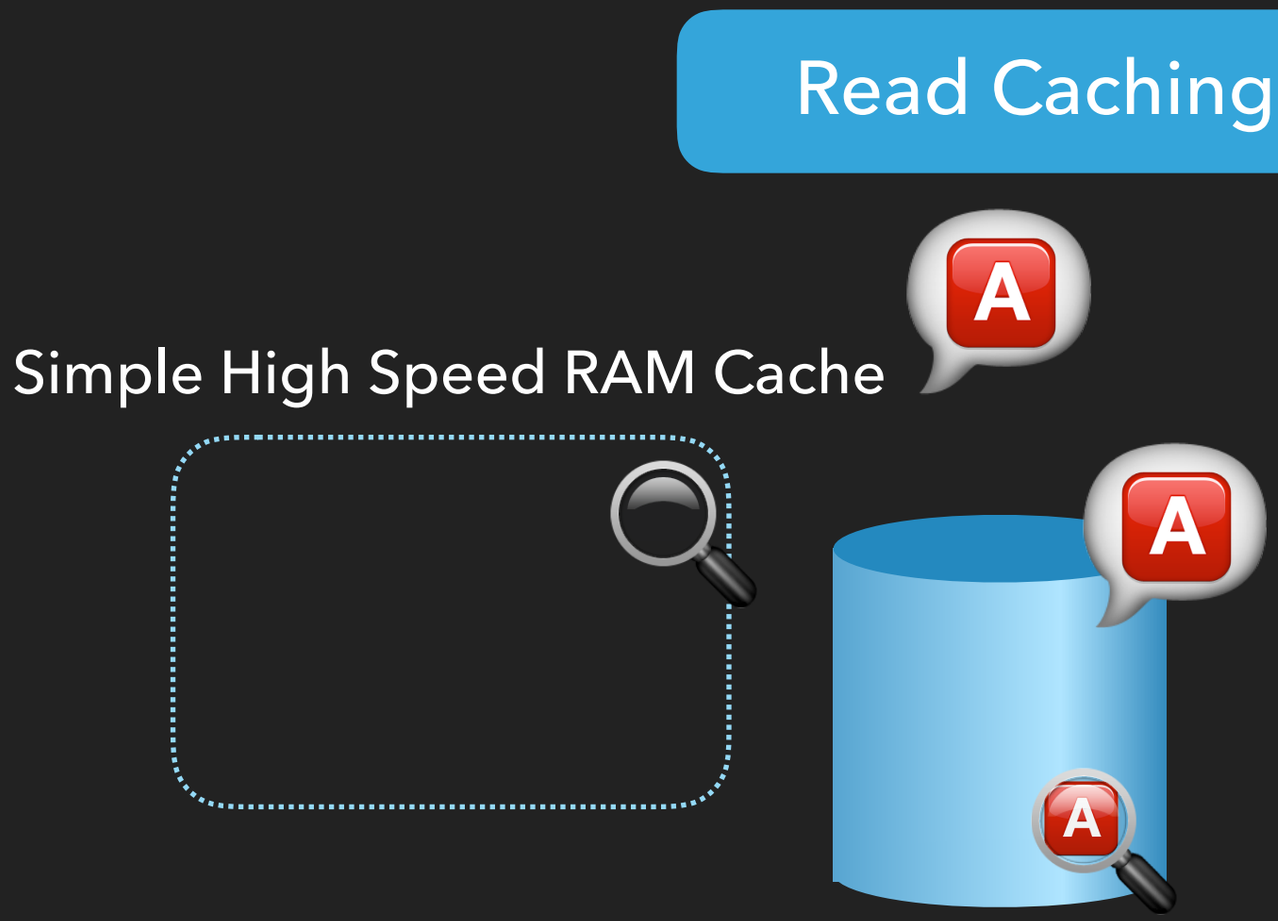


- Scales read transactions linearly with the number of agents
- Scales out read hits, read-misses & full table scans
- Frees up master database to handle writes
- Can segregate analytics queries to dedicated databases

- Must sacrifice either read consistency or write response times
- Writes must cross interconnect
- Write throughput is unimproved, and could be decreased

SCALING PRACTICES

Read Caching

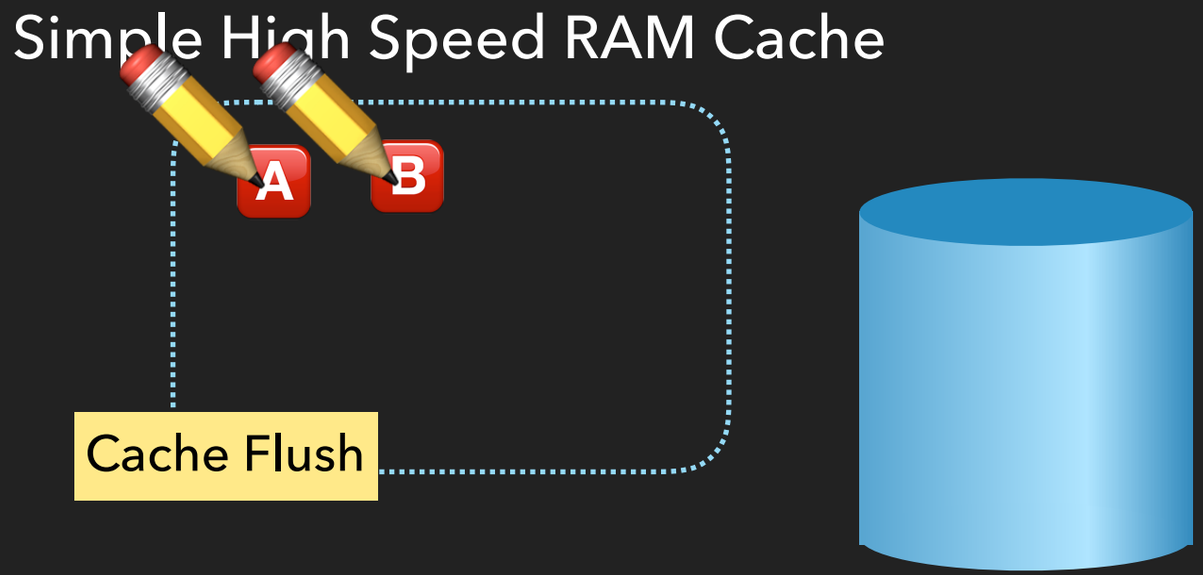


- Can have a huge positive impact on read response times
- Frees up database to handle writes, read-misses and full-table scans

- Relies on repeated access, random workloads will see less benefits
- No benefit to large read throughput
- Only benefits reads
- Sacrifices consistency – Now the developer needs to handle caching

SCALING PRACTICES

Write Coalescence



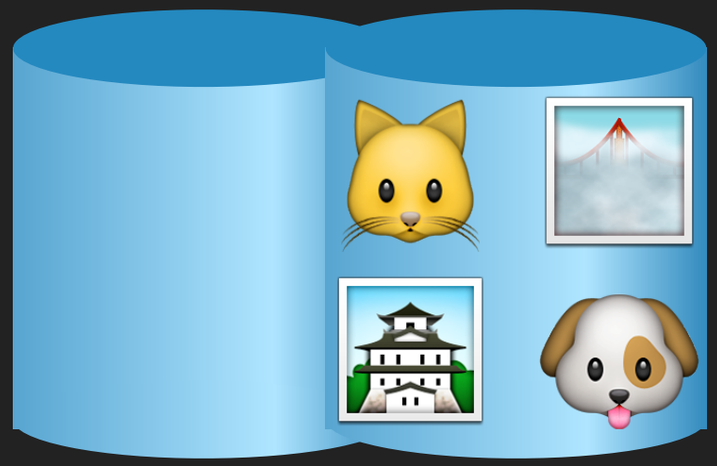
- Can have a huge positive impact on write response times and throughput
- Frees up database to handle reads, read-misses and full-table scans
- Only benefits writes
- Heavily sacrifices consistency – Now the developer needs to handle very complex caching

> THERE ARE ONLY TWO HARD THINGS IN COMPUTER SCIENCE: CACHE INVALIDATION AND NAMING THINGS.
>
> Phil Karlton

SCALING PRACTICES

Vertical Partitioning

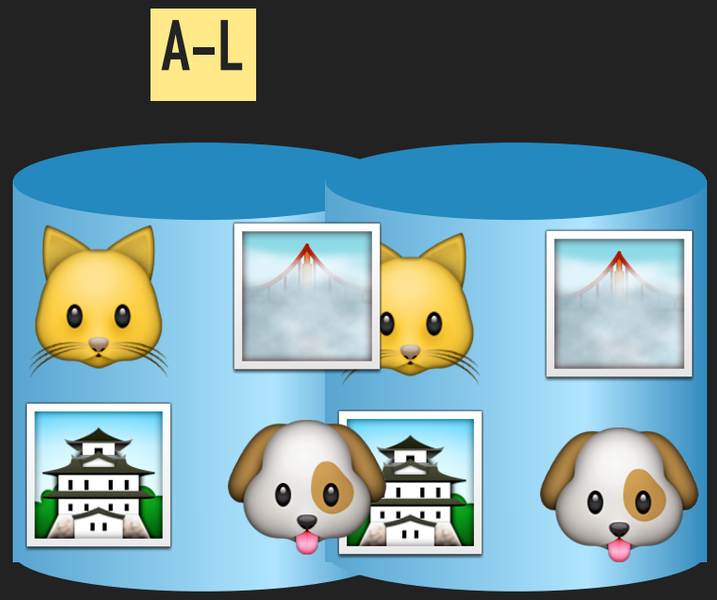
- Smaller segregated datasets carry tons of advantages
- Related entities can be co-located onto the same node.
- This also relates easily to the micro-services trend



- Highly dependent on the data model of your application
- Weaker consistency & transactions between partitions
- Joins that cross partitions will incur interconnect latency
- Trying to force vertical partitioning where it doesn't make sense will create massive headaches for joining.

SCALING PRACTICES

Horizontal Partitioning



- Partitions also called “shards”.
- Can scale any aspect: connections, read hits, read misses, writes, analysis
- Independent of data model

- Dependent on level of joining
- Can introduce weaker consistency & transactions
- Joins may cross partitions, introducing interconnect latency
- Bulk commits may introduce new complexity for carrying across partitions

ORACLE

- Connection scaling through Oracle *Real Application Cluster* (RAC)
- Master/Agent Replication through Data Guard or Golden Gate
- Partitioning & caching are left to the application

MYSQL

- Native Master/Agent replication
- Horizontal partitioning available with Galera Project (not represented here)
- Vertical partitioning & caching are left to the application

MEMCACHE

- Supports write coalescing & LRU read-through caching
- Primarily in-memory with write-backs/flushing support
- No support for relational features & limited transactions

POSTGRESQL

- Native Master/Agent replication
- Support for relational & non-relational data
- Partitioning & Caching are left to the application

MYSQL

ORACLE

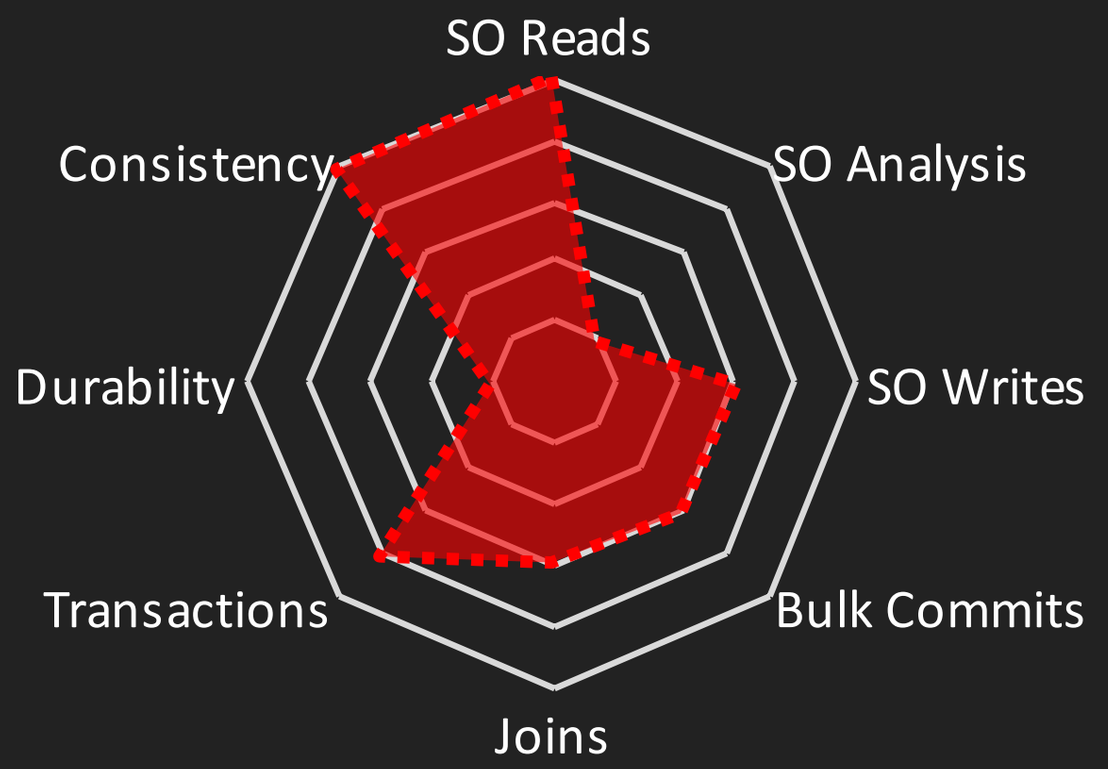
POSTGRESQL

MONGODB

- Implements native read-caching, write coalescence & horizontal partitioning
- Weak consistency, nearly no transactions, no relational features

GEMFIRE

- Primarily in-memory store with support for read-caching, write coalescence, horizontal & vertical partitioning
- Good support for transactions & joins, with special features to prevent interconnect latency



REDIS

- Primary in-memory key-value store
- Support for read-caching, write coalescence, master/agent replication
- Innovative support for complex values, joins & transactions

HDFS

- Unstructured data store with support for horizontal & vertical partitioning
- Joins data across shards using Map & Reduce semantics/libraries

GREENPLUM

- Relational OLAP database with row & column orientation & support for horizontal partitioning
- Joins data across shards using Map & Reduce semantics, with optimized queries that can, when needed, communicate across mappers

OPEN DISCUSSION HERE