NoSQL & Distributed Log

Diego Pacheco

About me...

- Cat's Father
- Principal Software Architect
- Agile Coach
- SOA/Microservices Expert
- DevOps Practitioner
- Speaker
- Author

diegopacheco

@diego_pacheco

http://diego-pacheco.blogspot.com.br/

https://diegopacheco.github.io/

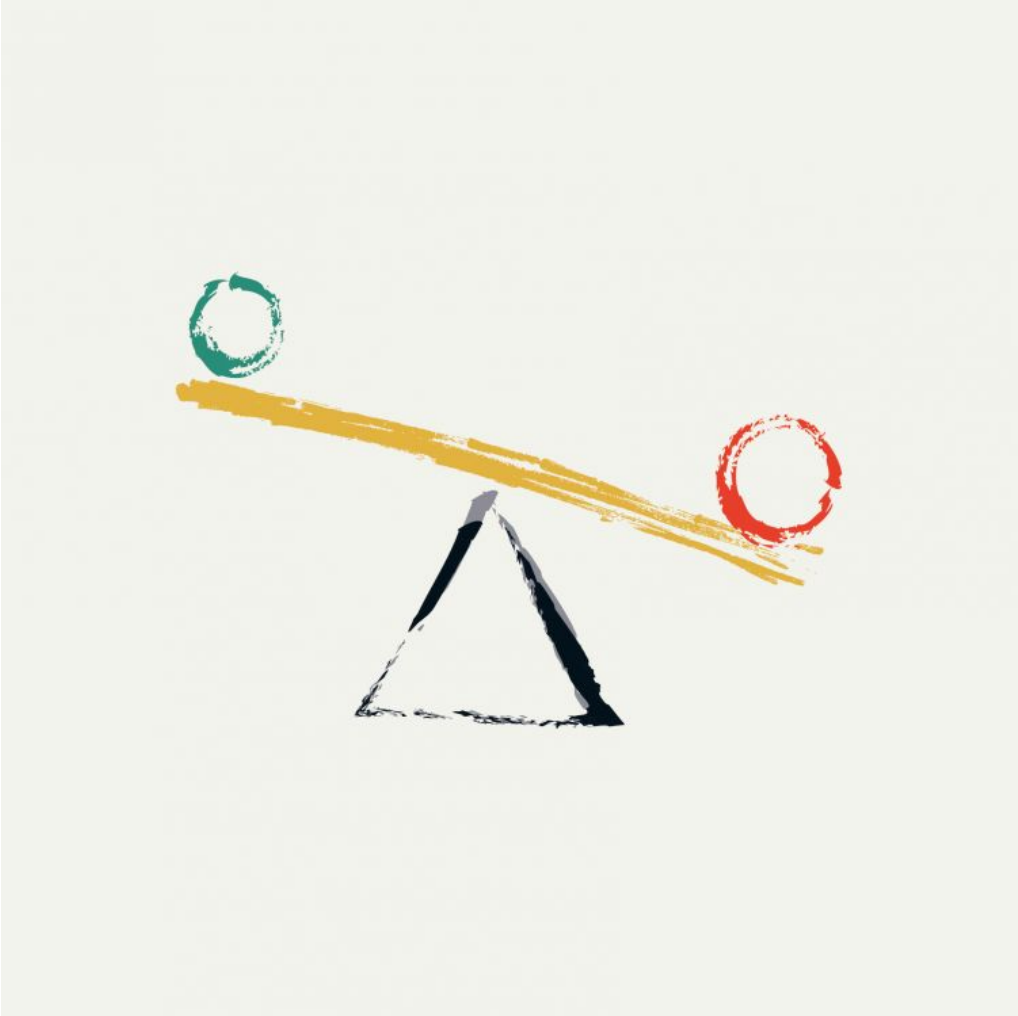
Design


Requirements

Going to the Beach "Metaphor"

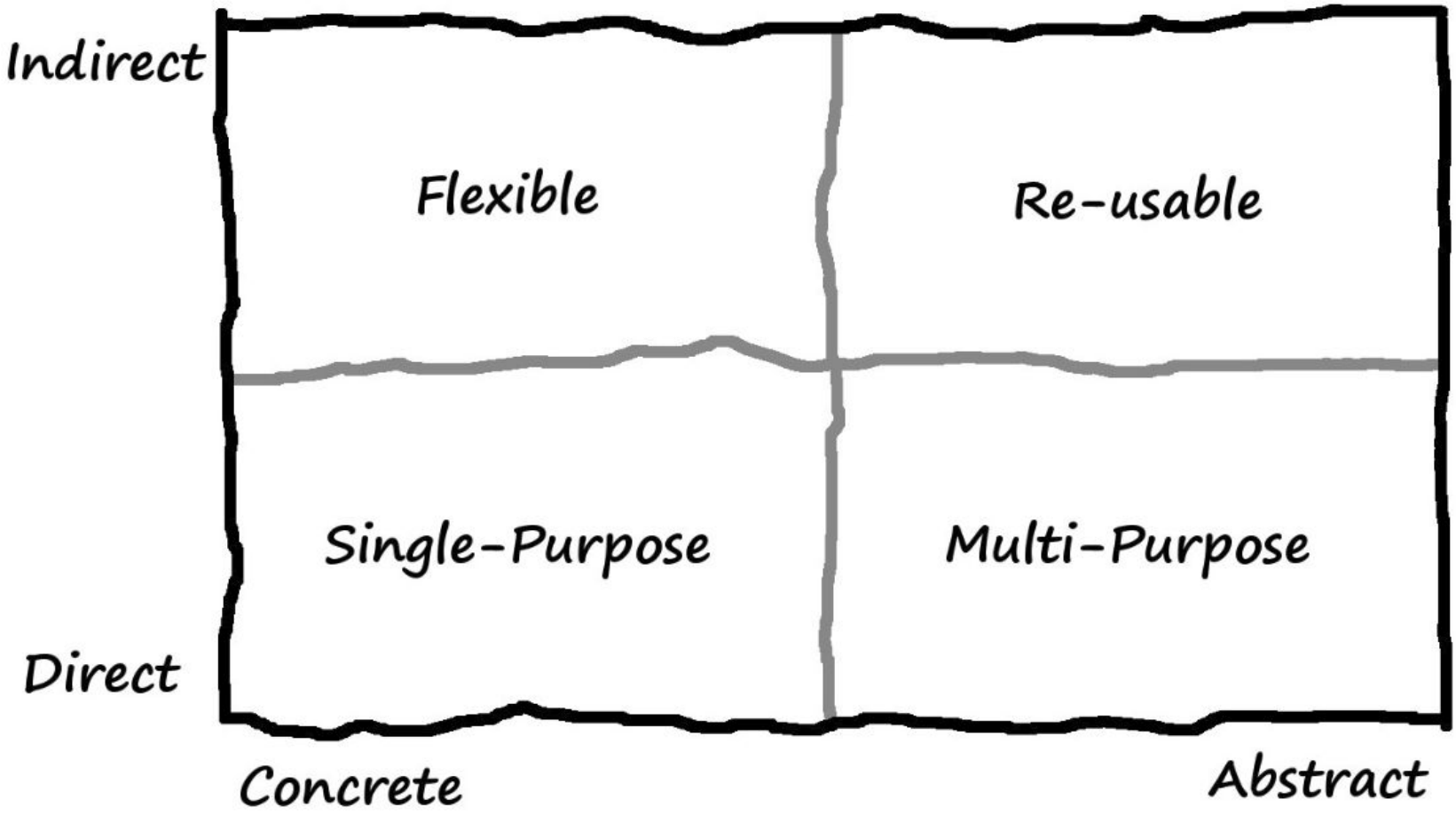
- Let's say dont have requirements about:
  - Beers
  - Corn
  - Surf
  - Fun
  - shower
- But you know we are going to a beach so even if anyone DID NOT asked you need to GET AHEAD and connect the dot.
- Architecture / Design are about being "flexible" enough to accommodate future possible changes.
- Not Necessary means CODING but for sure means thinking and being ready for.

Propósito: SOC



Architecture/Design is all about Tradeoffs

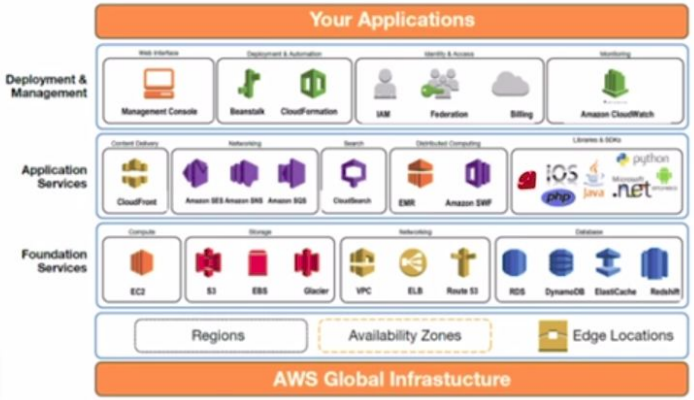
- Given a USE CASE:
  - What Benefits?
  - What issues it has?
  - What dependencies?
  - What risks?
  - How we improve it?
  - How we validate it?
  - How we know we are right?
- Build VS Buy Analysis
- Open Source VS Paid software
- How much up to front work?


Indirect
Flexible
Re-usable
Single-Purpose
Multi-Purpose
Direct
Concrete
Abstract

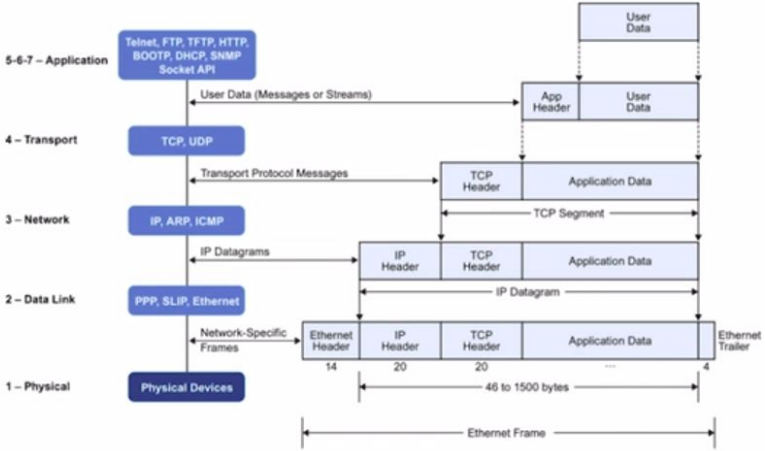
Engineering: approach by decomposition

1. Identify a problem
2. Break down a big problem to smaller problems
3. Design algorithms for each individual problem
4. Compose solutions into a system (get a “stack”)

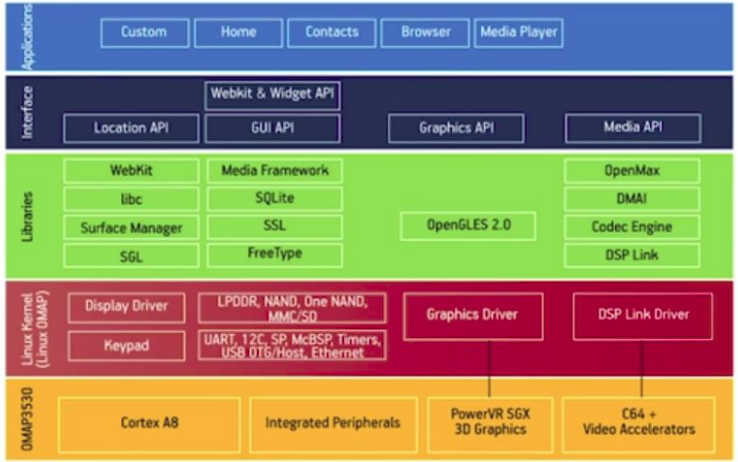
AWS stack



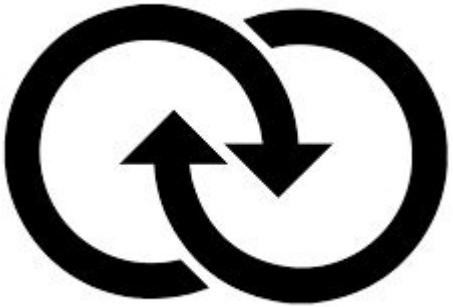
TCP/IP stack



Android software stack



SQL



The universe as we know...

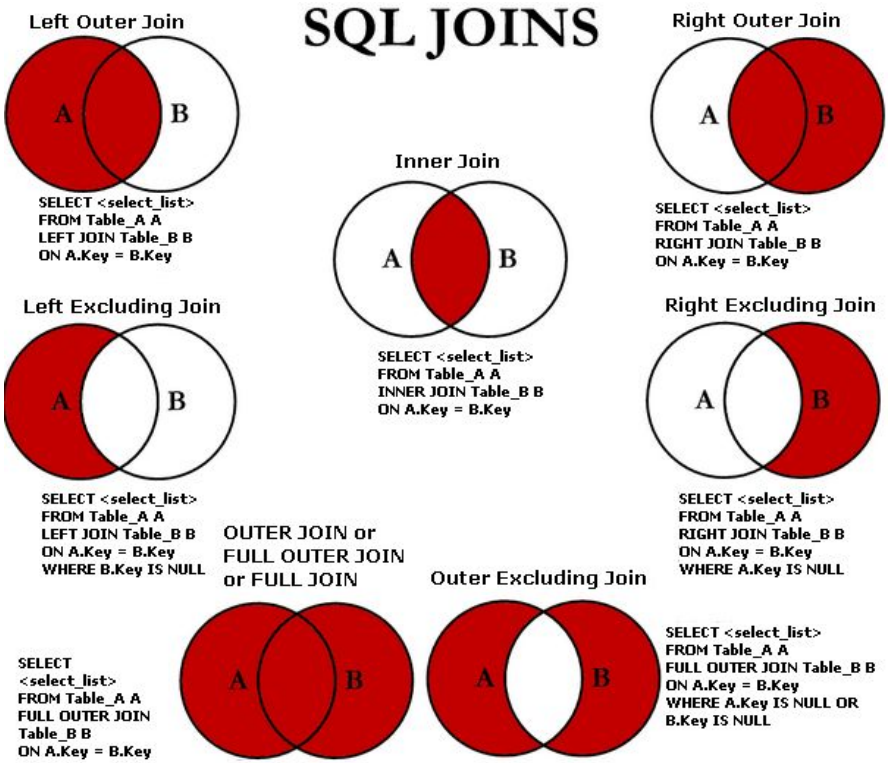

42

The universe as we know...

=

=

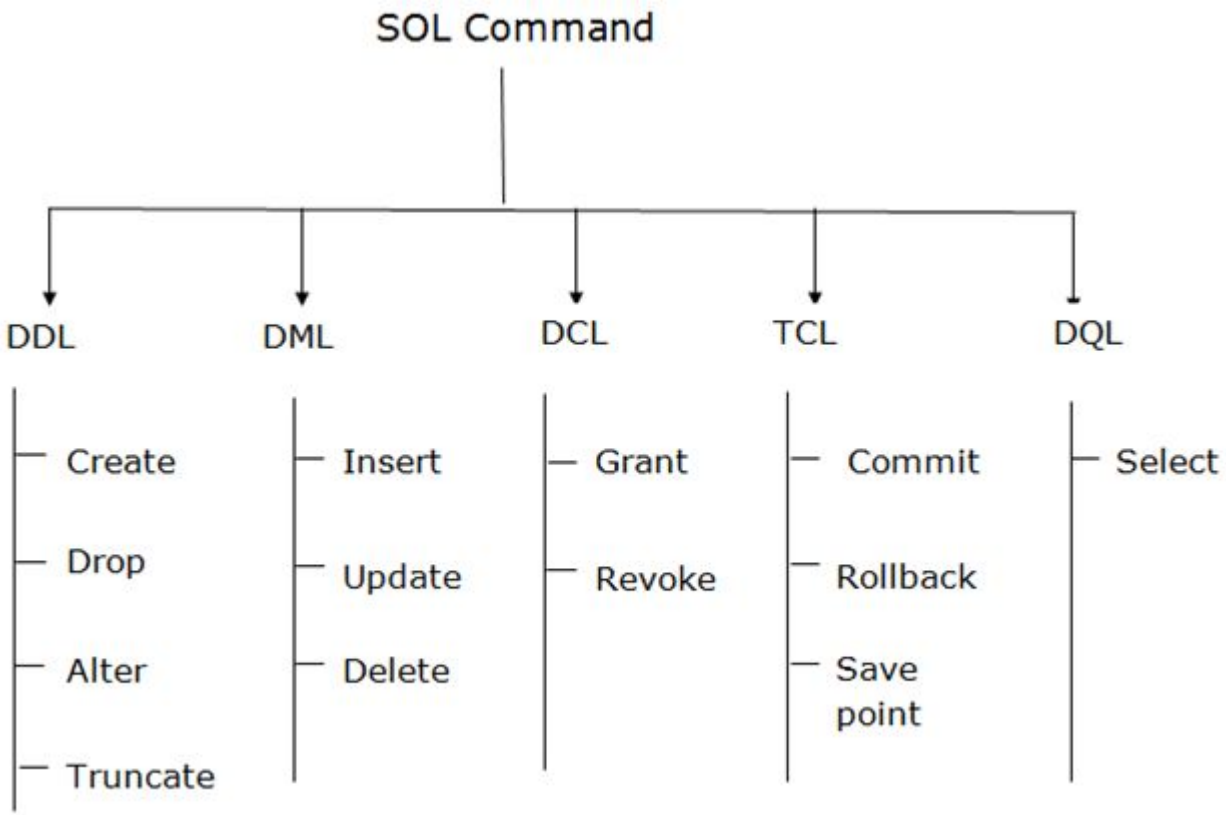
The universe as we know...



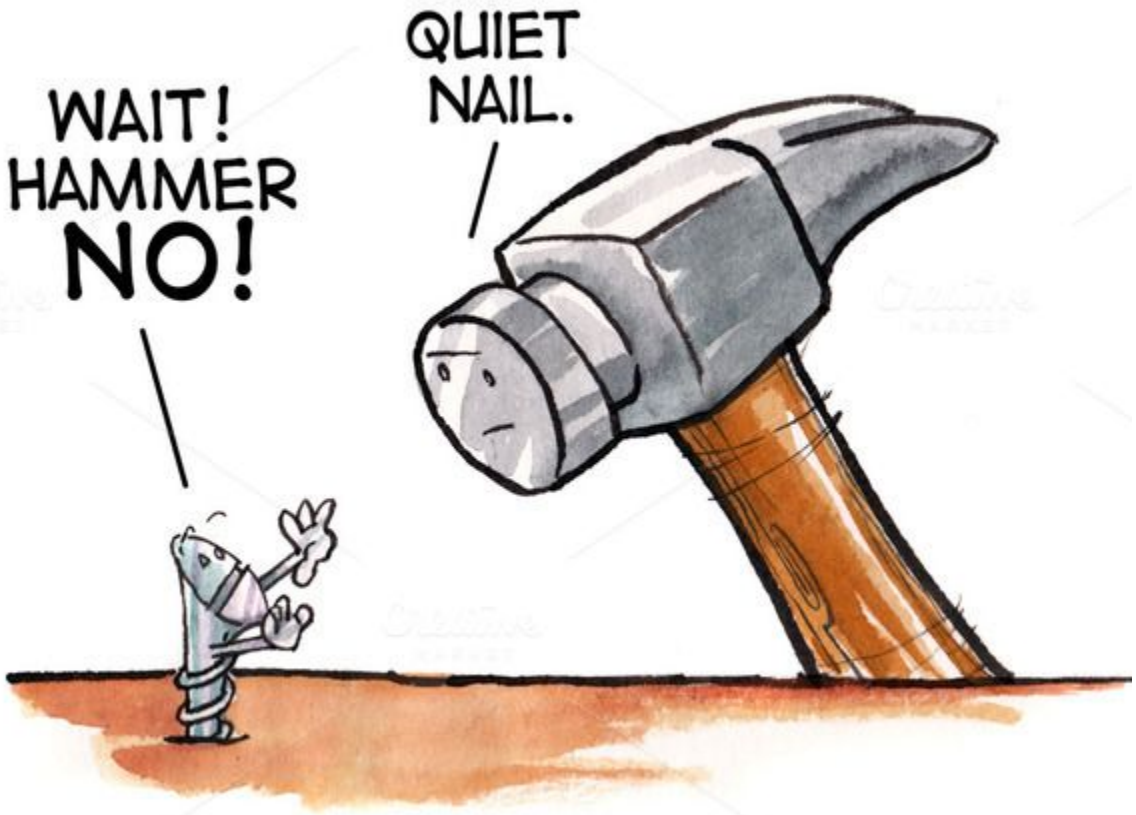
'A' & 'B' are two sets.

1. AnB = Inner Join('n' - intersection)
2. Au(AnB) = Left Join ('u' - Union)
3. (AnB)uB = Rigjt Join
4. AuBu(AnB) = Outer Join
5. A- B = Left Join Excluding Inner Join or Relative Component
6. B- A = Right Join Excluding Ineer Join
7. (A-B)u(B-A) = Outer Join Excluding Inner Join

Declarative: I don't care, just get it done....



Relational abuse !!!



NoSQL

NO
SQL

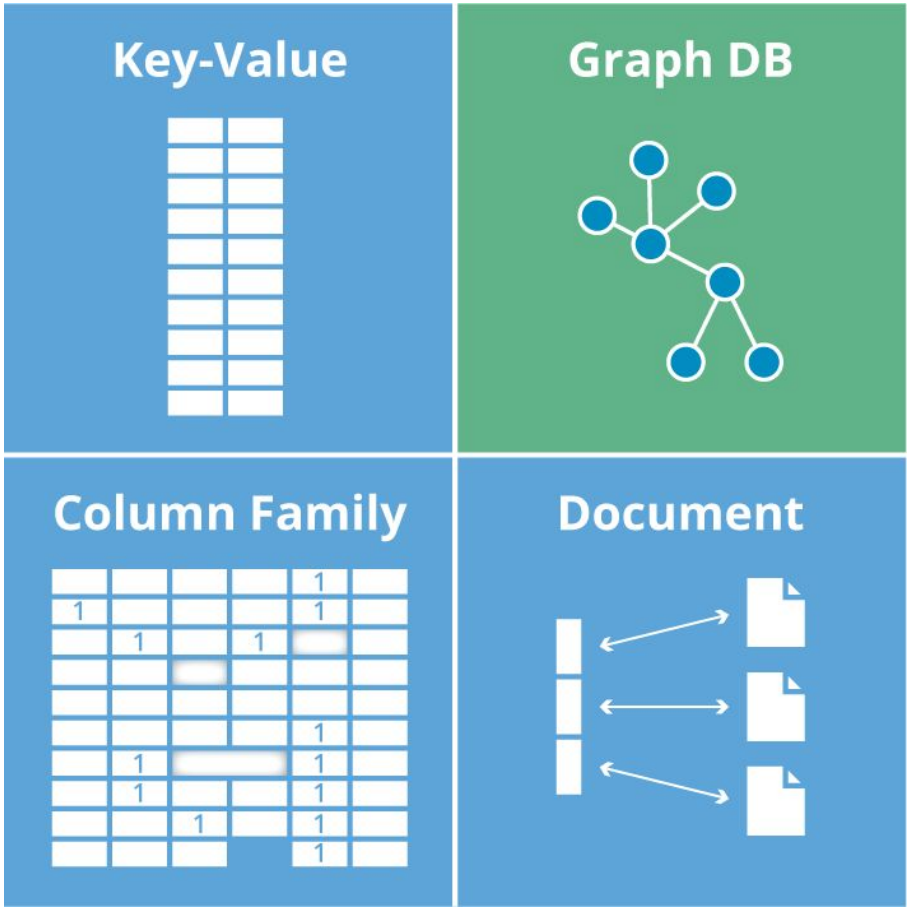
Much Bigger Universe...

a Different universe...

- What is NoSQL about?
  - High Scalability
  - Low Latency
  - High Throughput
  - High Availability (IF "AP" system)
  - Fast Performance
  - Linear Scalability
  - Better Design Options
  - Great for Big Data
  - Easier Replication (less schema constraints)
  - Relation DB striped in several parts - Application and Ops do more.
  - No Free Lunch: Operation: Several times is hard

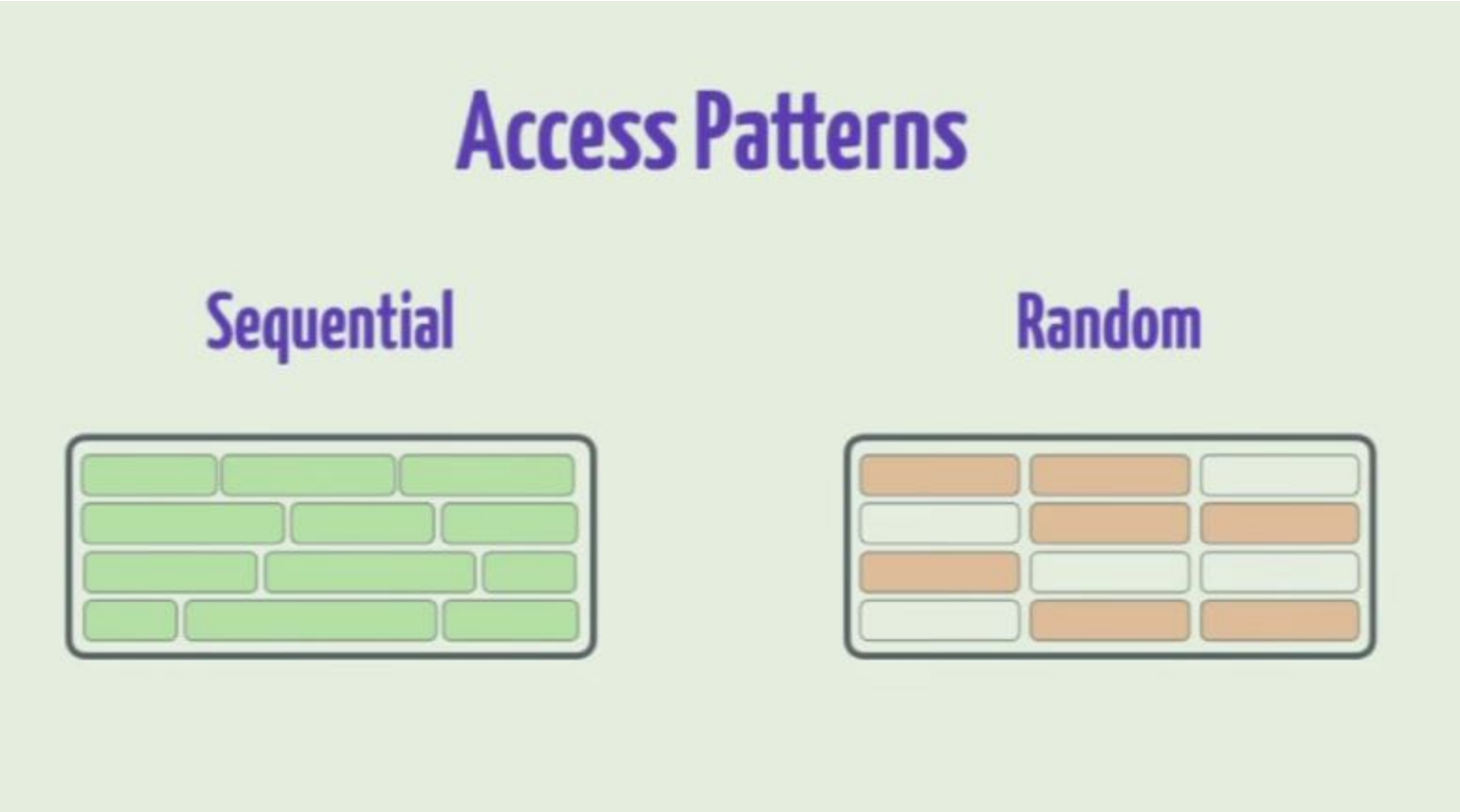
NO SQL

Different ways do ASK and ANSWER questions....

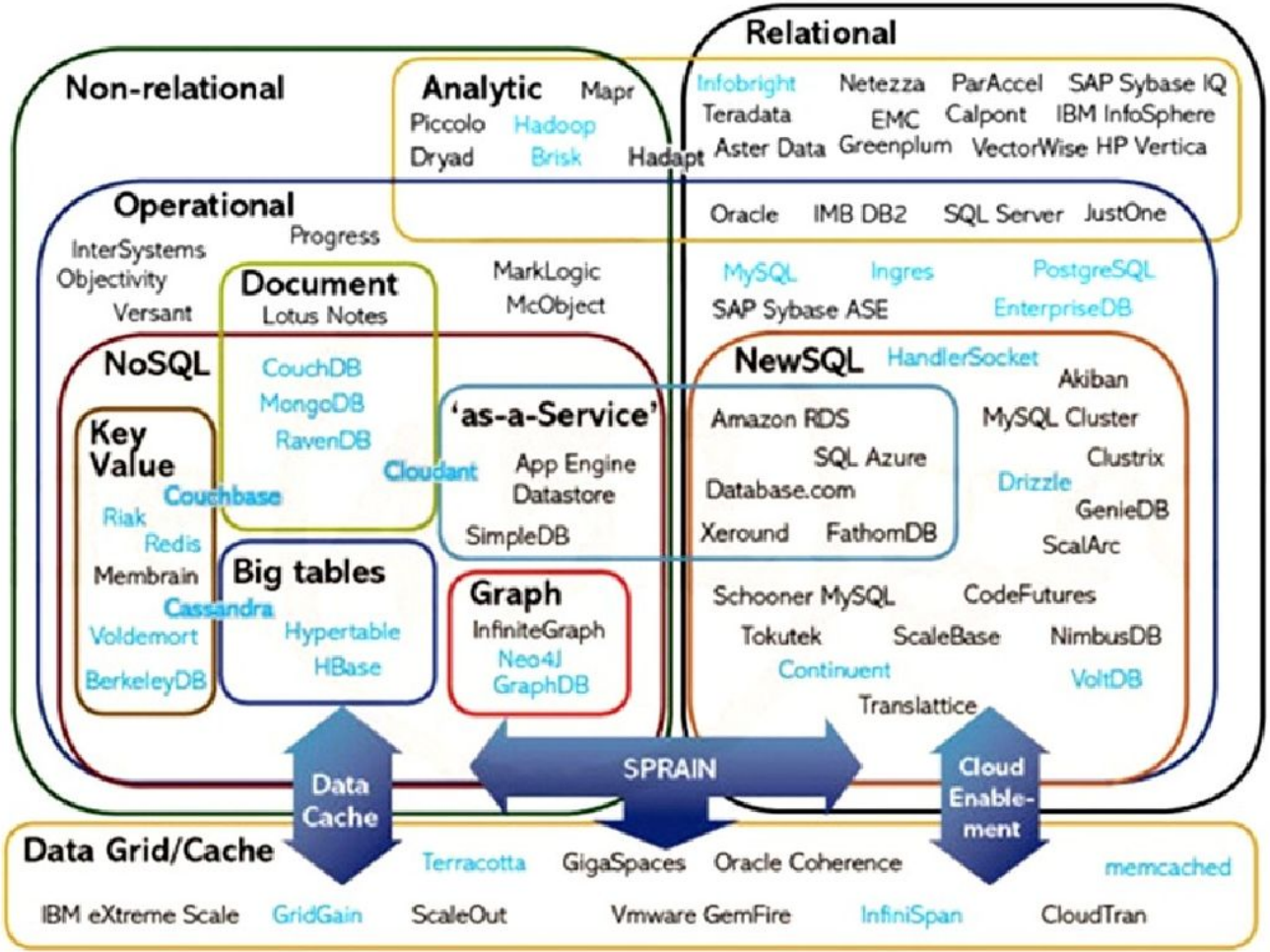


- <u>NoSQL WAY</u>
  - "No Tables"
  - "No Joins"
  - "No ACID"
  - Different "schema"
- More Data Structures
- I need to know what I'm m doing.
- What do I want todo?
- What Access patterns do I have?
- I need fix problem in "parts".
- Looks like much more work... (yes)
- ... Much better Performance / Scalability
- ... "RIGHT DESIGN"
- What about integrity?
- Be my guest ! LOL
- Looks like we "strip" the "DB" and we are doing all the work on APP side? YES!
- Seriously? YES!

NoSQL is about Access Patterns. What queries / questions you need to answer?




OH JEEZ!
OH MAN!!


Relational
Non-relational
Analytic
Mapr
Piccolo
Hadoop
Dryad
Brisk
Hadapt
Infobright
Netezza
ParAccel
SAP Sybase IQ
Teradata
EMC
Calpont
IBM InfoSphere
Aster Data
Greenplum
VectorWise
HP Vertica
Operational
Oracle
IMB DB2
SQL Server
JustOne
InterSystems
Progress
Objectivity
Versant
Document
Lotus Notes
MarkLogic
McObject
MySQL
Ingres
PostgreSQL
SAP Sybase ASE
EnterpriseDB
NoSQL
CouchDB
MongoDB
RavenDB
Cloudant
Key Value
Couchbase
Riak
Redis
Membrain
Cassandra
Voldemort
BerkeleyDB
Big tables
Hypertable
HBase
'as-a-Service'
App Engine
Datastore
SimpleDB
Graph
InfiniteGraph
Neo4J
GraphDB
NewSQL
HandlerSocket
Akiban
Amazon RDS
SQL Azure
Database.com
Xeround
FathomDB
MySQL Cluster
Clustrix
Drizzle
GenieDB
ScalArc
Schooner MySQL
CodeFutures
Tokutek
ScaleBase
NimbusDB
Continuent
VoltDB
Translattice
Data Cache
SPRAIN
Cloud Enable-ment
Data Grid/Cache
Terracotta
GigaSpaces
Oracle Coherence
memcached
IBM eXtreme Scale
GridGain
ScaleOut
Vmware GemFire
InfiniSpan
CloudTran


OH JEEZ!
OH MAN!!

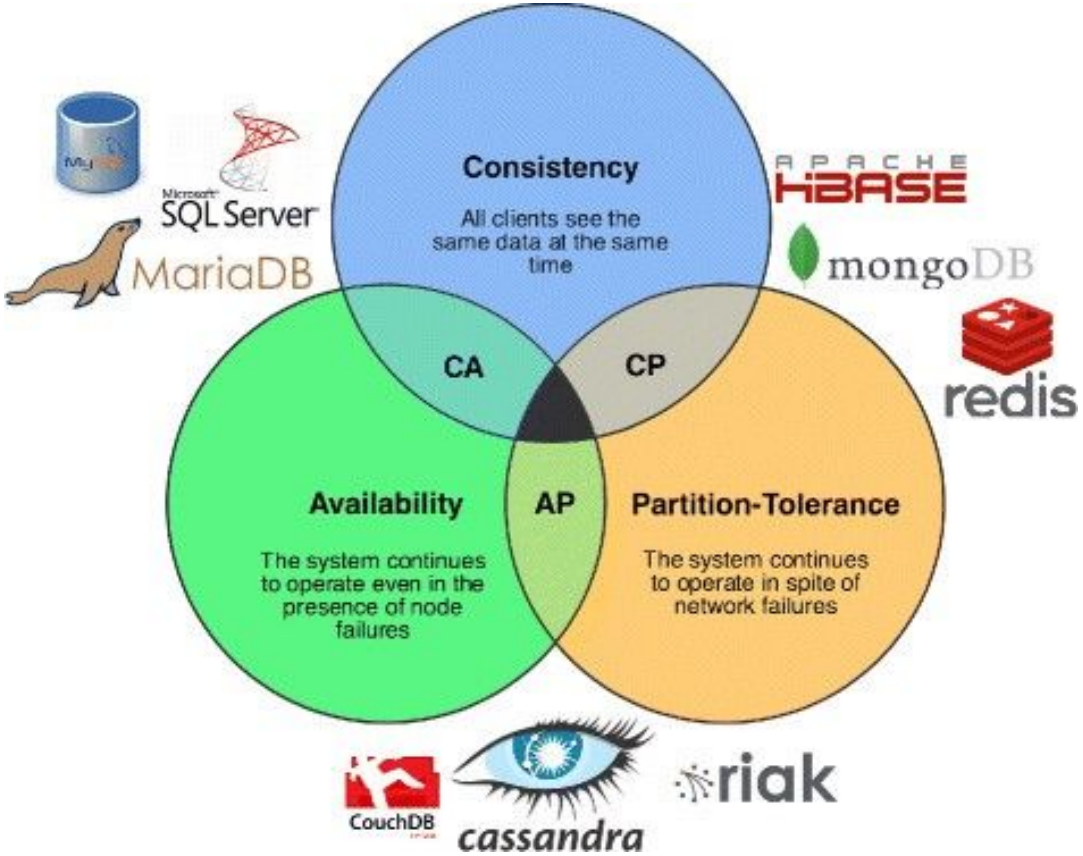
Philip Schwarz @philip_schwarz · Feb 26

Replying to @unclebobmartin @sarahmei @holly_cummins

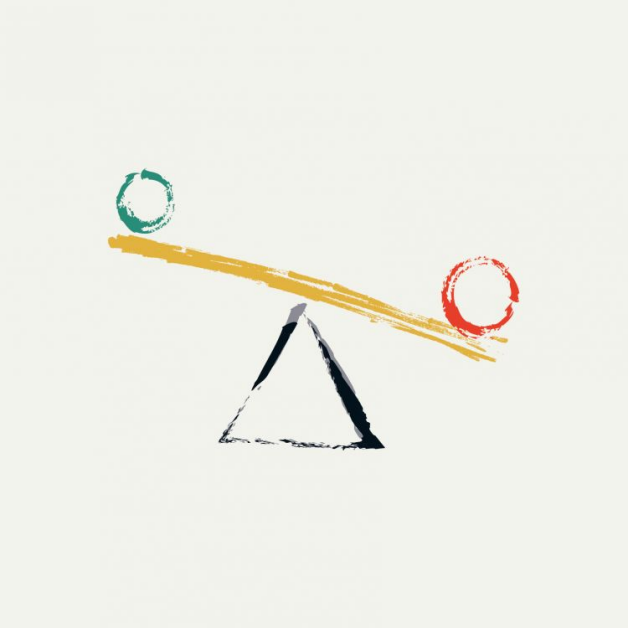
the first time I came across 'typing is not the bottleneck' was in @GeePawHill's 2009 post: anarchycreek.com/2009/05/26/how… - @sbastn then created the monkey stickers web.archive.org/web/2009093011…

1 2 5

CAP: Pick 1, forget the P.



Type	Example
Key-Value Store	redis, riak
Wide Column Store	HBASE, cassandra
Document Store	mongoDB, CouchDB relax
Graph Store	Neo4j, InfiniteGraph The Distributed Graph Database

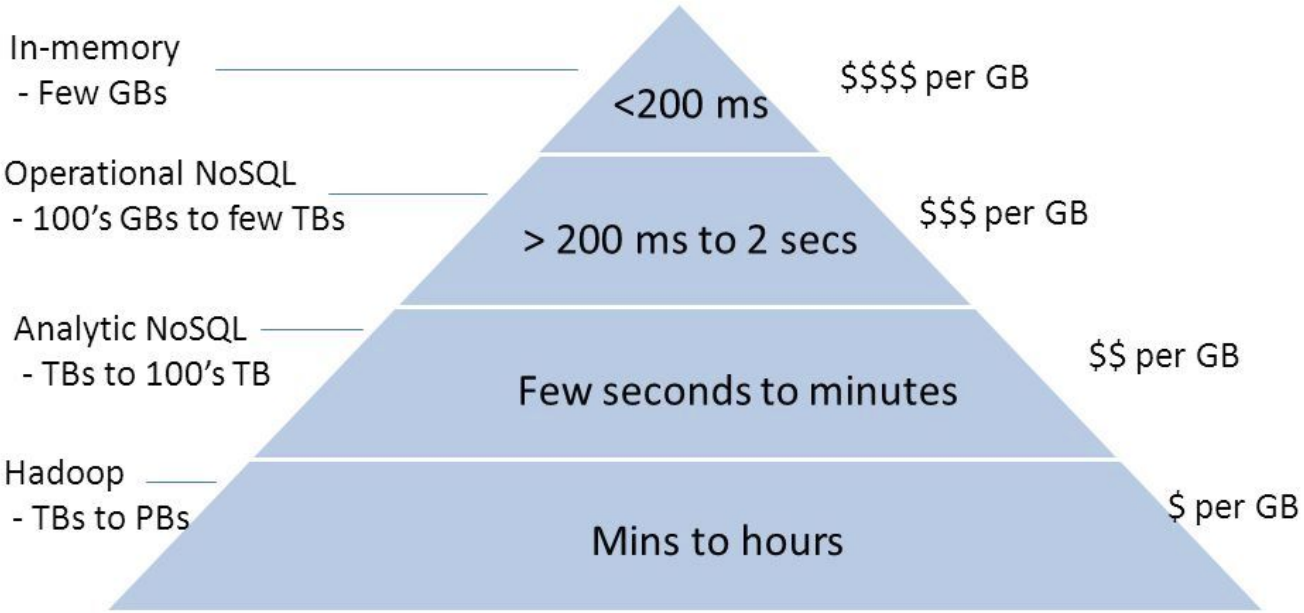


Low Level in Detail

Trade-offs

* Data Structures
* Memory / Disk
* IO / Storage
* Monolith DB vs The Log

Workload Economics



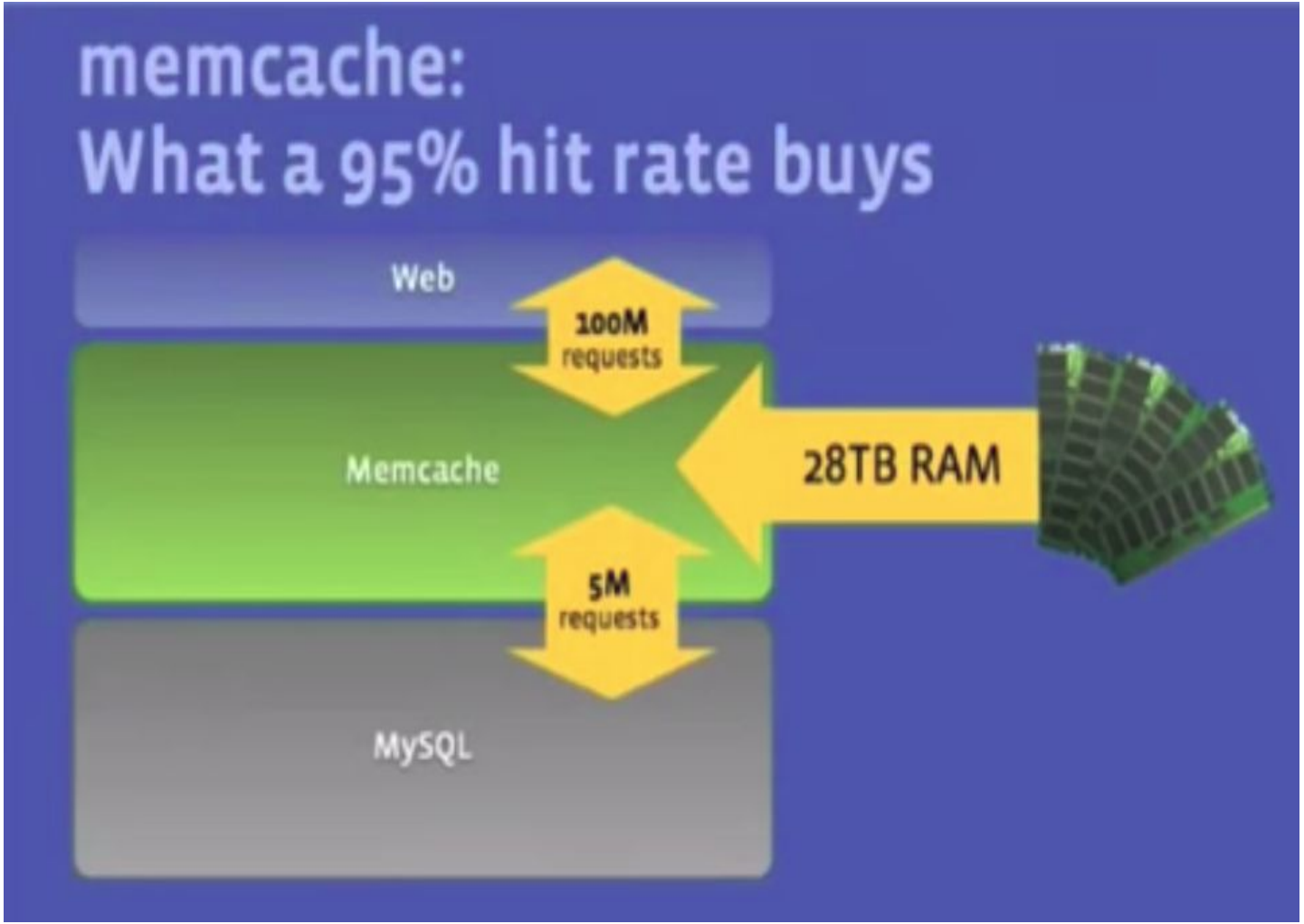
* Nieman Marcus approach presented at TDWI Solution Summit 2014

How many copies is enough?

More copies,
More resilient

Fewer copies,
Faster replication

We refer to any group of nodes which is ‘enough’ as a **replication quorum**



In 2013 Facebook was scaling pretty much based on Memcached which is a in-memory NoSQL memory database.

Reduce number of DB Calls (100M -> 5M)

28 TB of RAM and +800 servers was used by Facebook in 2013 .

Disk DS Trade-offs

On Disk Data Structures

Mutable

- Space allocation
- In-place updates
- Fragmentation

Immutable

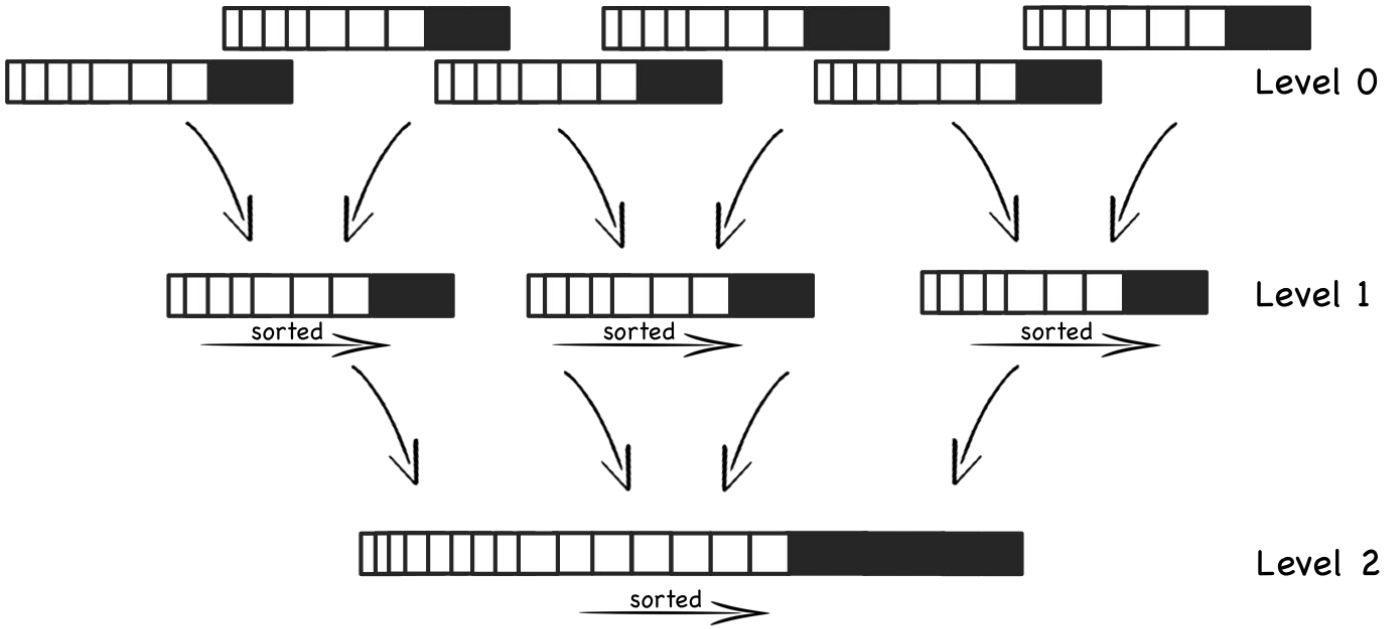
- Sequential writes
- Multiple read sources
- Needs merge

Relation Data Structure = B+ Trees

Summary

- Mutable
- Read-optimised
- Requires splits/merges/rebalancing
- Block storage optimised
- Overhead for in-place updates

Data Structures: LSM Trees

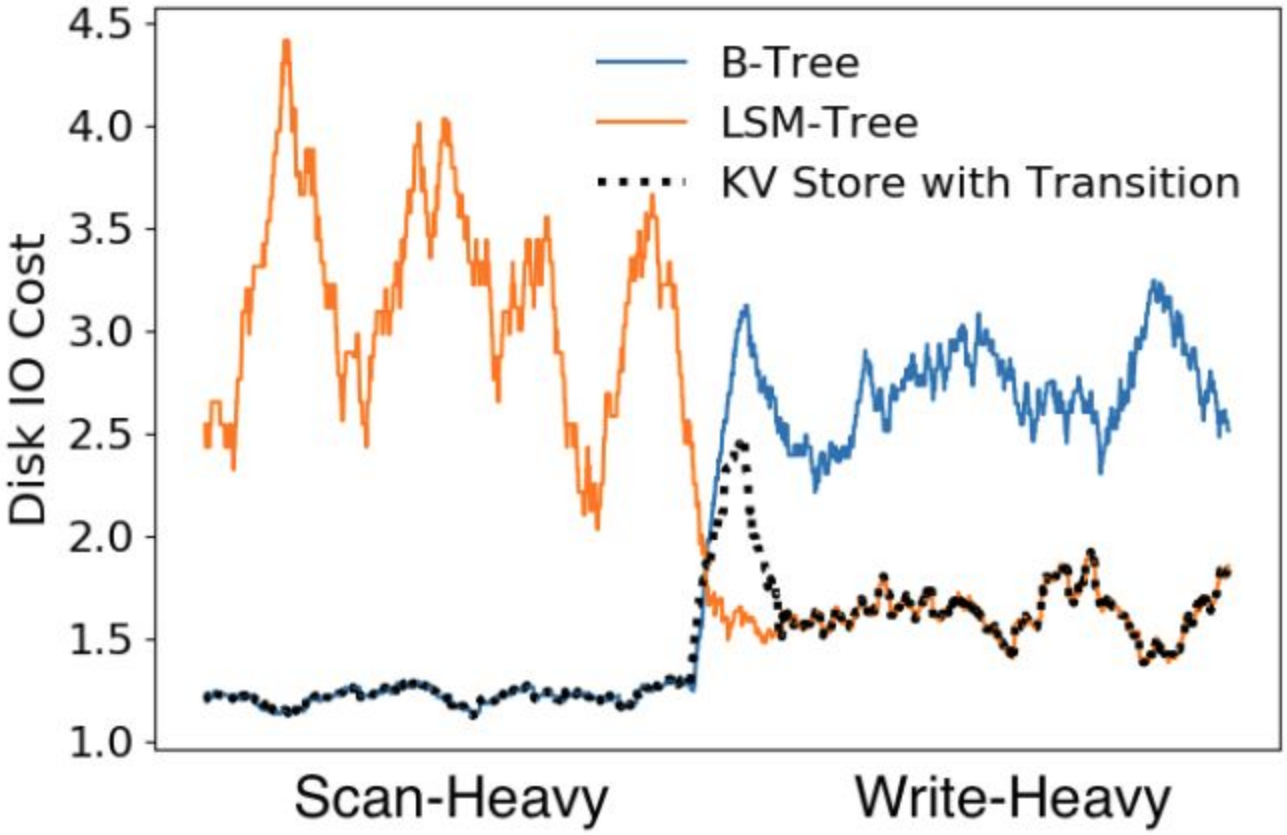


Compaction continues creating fewer, larger and larger files

NoSQL Data Structure = LSM Trees

Summary

- Immutable
- Write-optimised
- Read multiplexing
- High maintenance costs
- Well-suited for concurrent ops
- Simple to implement


B-Tree
LSM-Tree
KV Store with Transition
Disk IO Cost
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Scan-Heavy
Write-Heavy

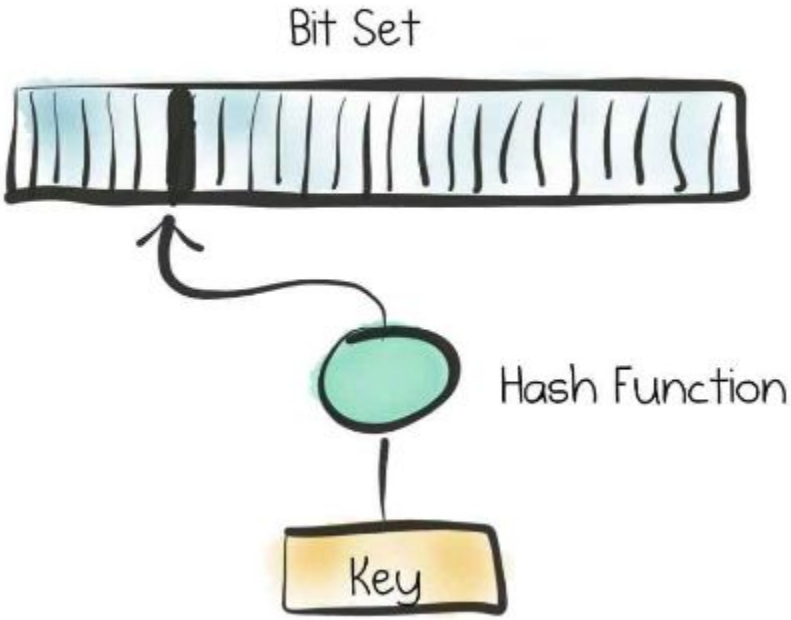
Bloom Filters

Bloom Filters

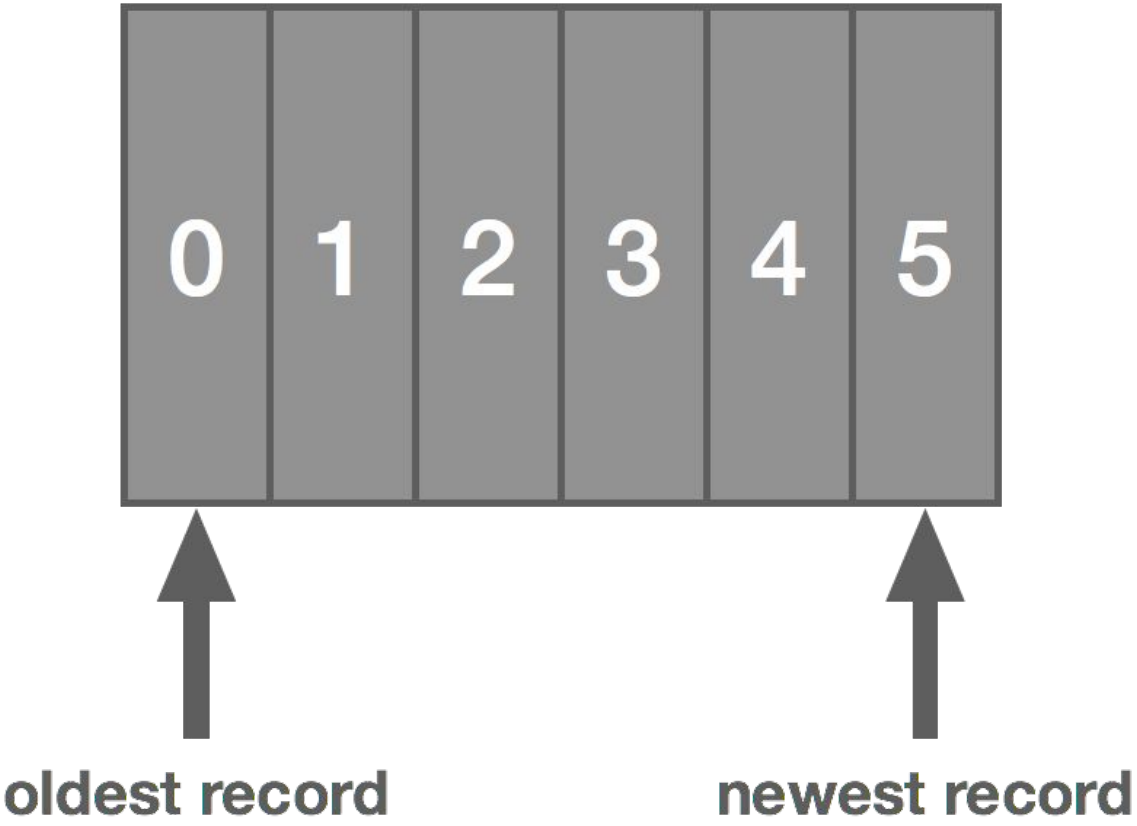
Answers the question:

Do I need to look in this file to find the value for this key?

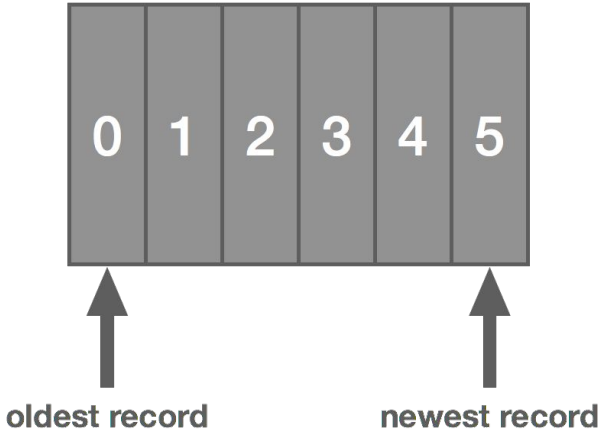
Size -> probability of false positive



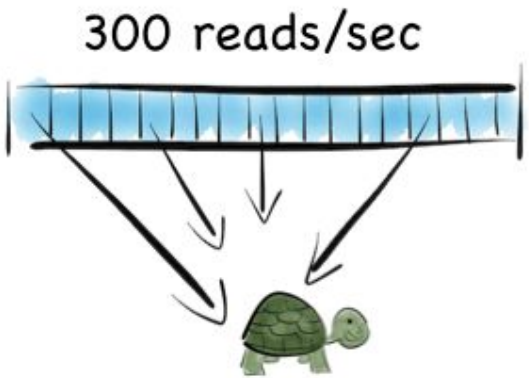
Distributed Logs

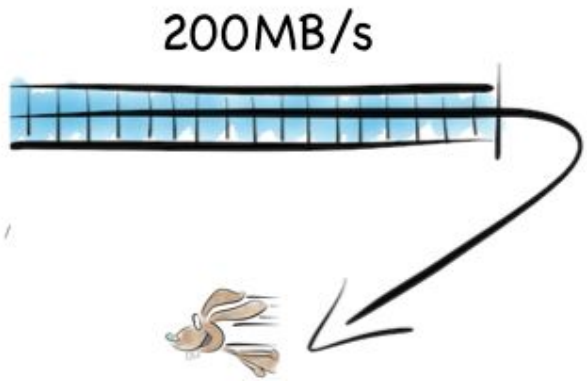


- What is a Log?
  - Simple "record"
  - Ordered Sequence of Immutable Event
  - Needed for Data Intensive "Big" Systems.
  - System using it:
    - Kafka
    - Amazon Aurora
    - Raft Protocol
    - Relational DBs (for Durability/Recovery) "redo" logs
  - Who does use it?
    - LinkedIN, Facebook, Twitter, Netflix, Apple, Google, Uber, Lyft, Spotify...



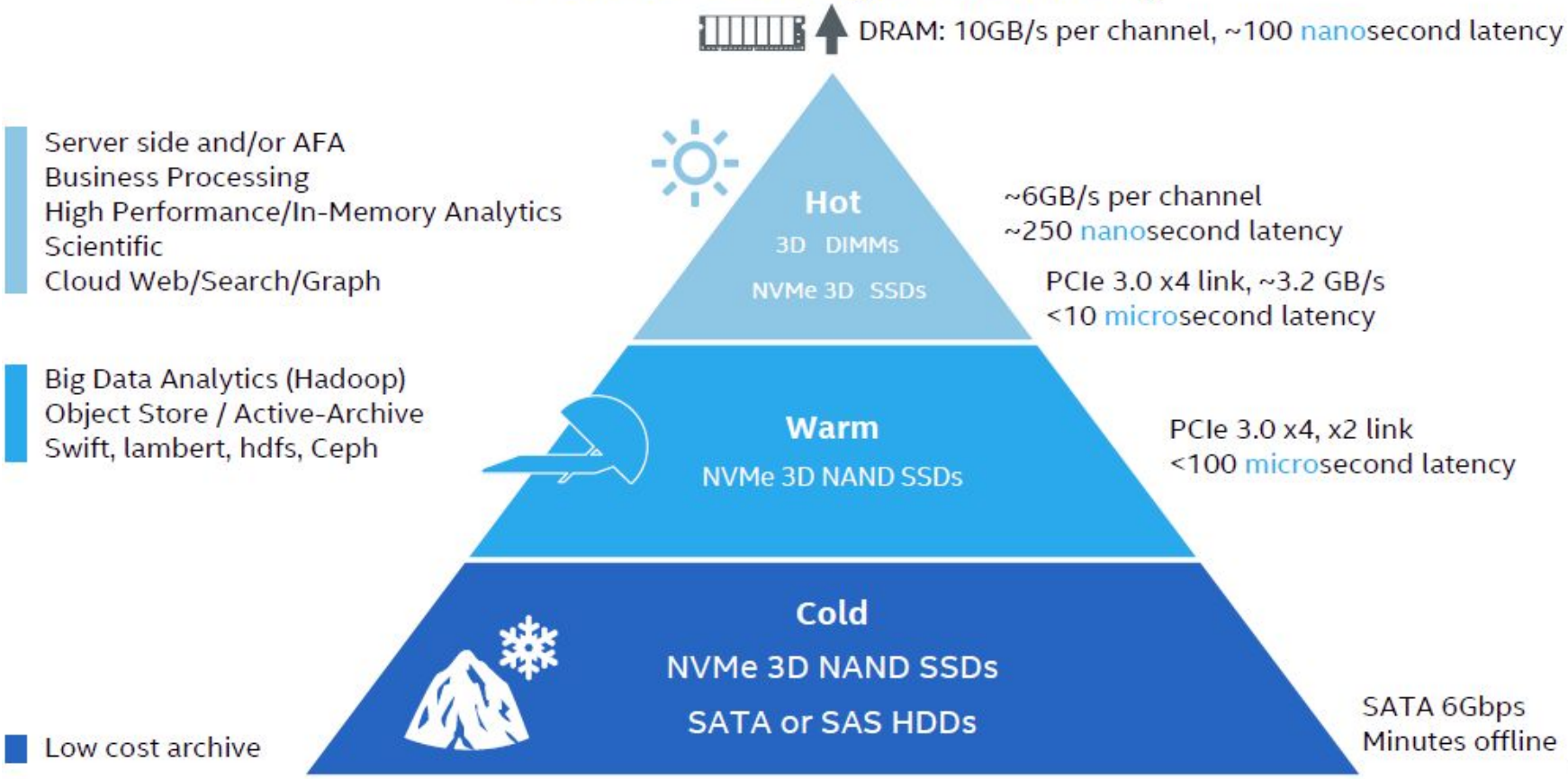
Random vs. Sequential Addressing





e.g. sequential is ~7000x faster for 100B rows

Future Storage Hierarchy



Comparisons between memory technologies based on in-market product specifications and internal Intel specifications.

Source: Intel - IDF2015


PRINCIPLES
VALUE
STRATEGY
STATEMENT
MISSION
INTEGRITY
CONDUCT
INNOVATION
STATEMENT
BUSINESS
COMPANY
PLAN
OBJECTIVES
IDEOLOGY
INNOVATION
CUSTOMER
ACCOUNT
VISION
EMPLOYEES
EXPENSE
PRICE
GUIDING
RULE
VISION
COMMERCE
GOALS
EXCELLENCE

No Places, RAW Events.

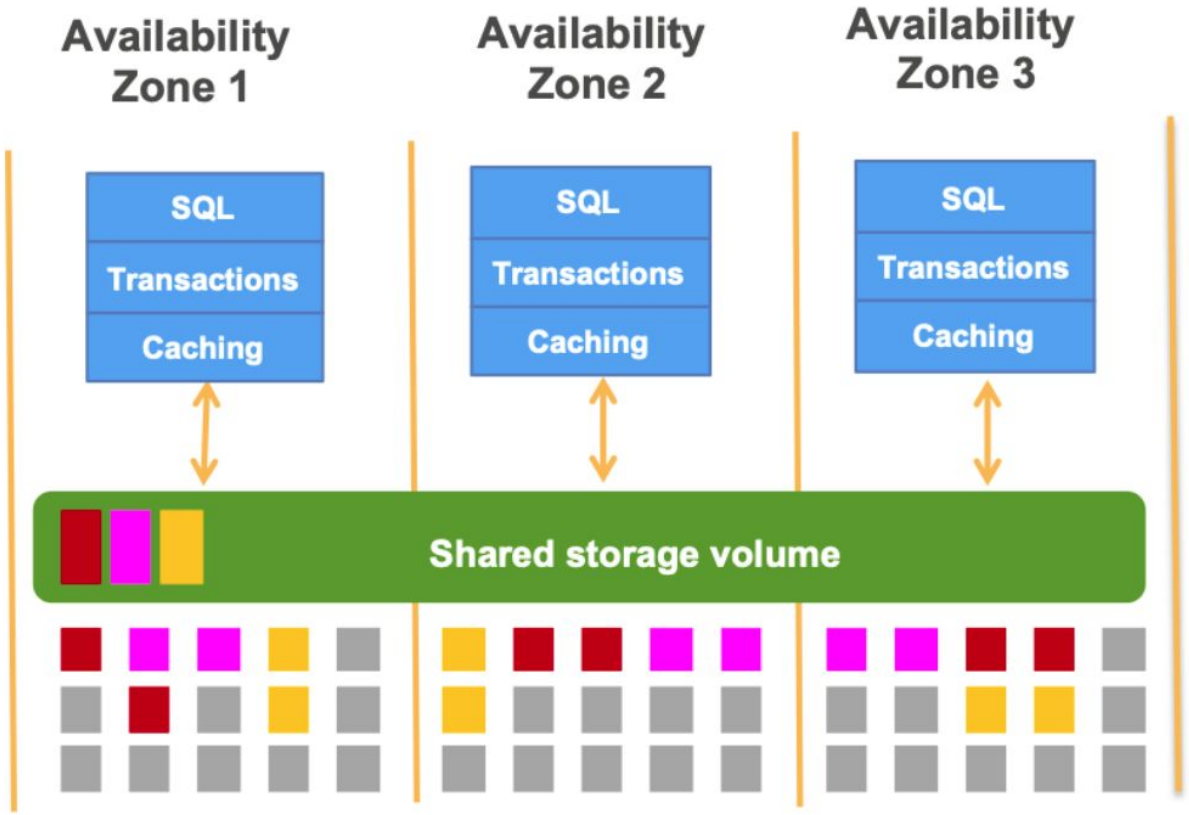
Immutability == Append ONLY

Segregation == Scalability

AWs Aurora Serverless Database == Modern, Well Engineered, totally based in Distributed Logs. Re-build the DB by the LOG, "The Log is the Database == Source of Truth".

Amazon Aurora

AWs Aurora Serverless Database == "The log is the Database".



More Throughput Thanks to the Log. Log is out from DB engine.

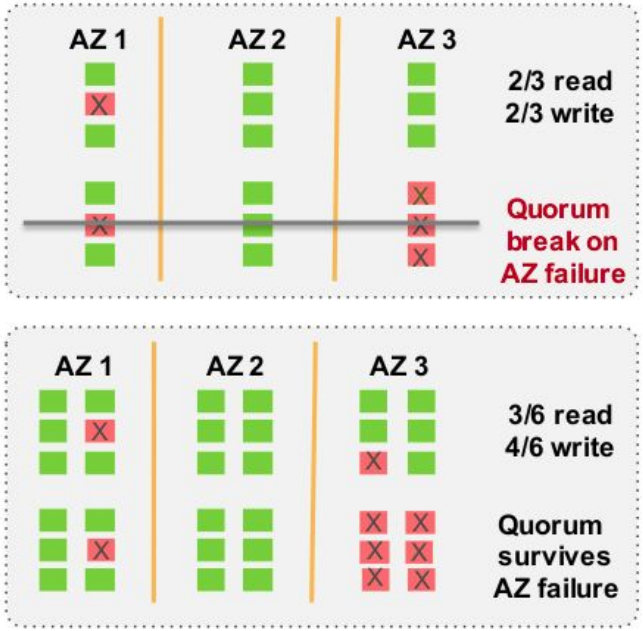


Figure 1: Why are 6 copies necessary ?

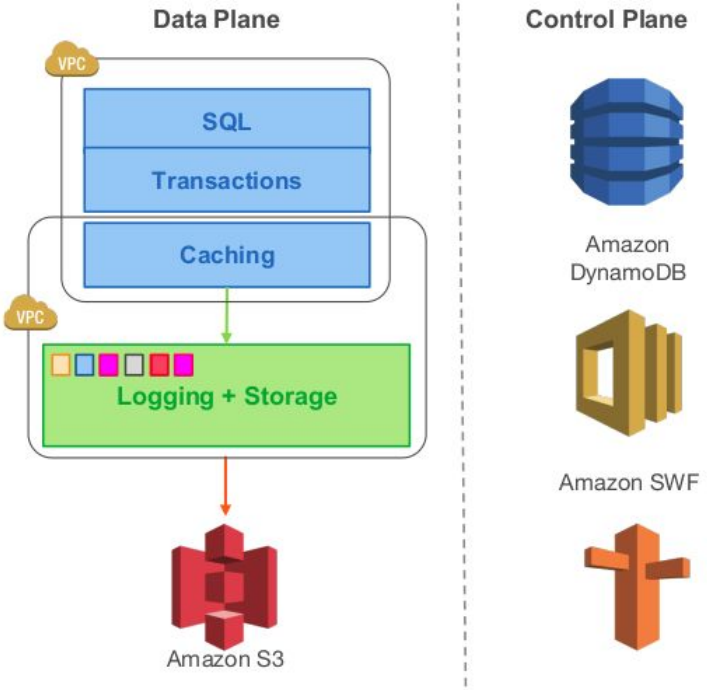


Figure 1: Move logging and storage off the database engine

Distributed writes - thanks to the log.

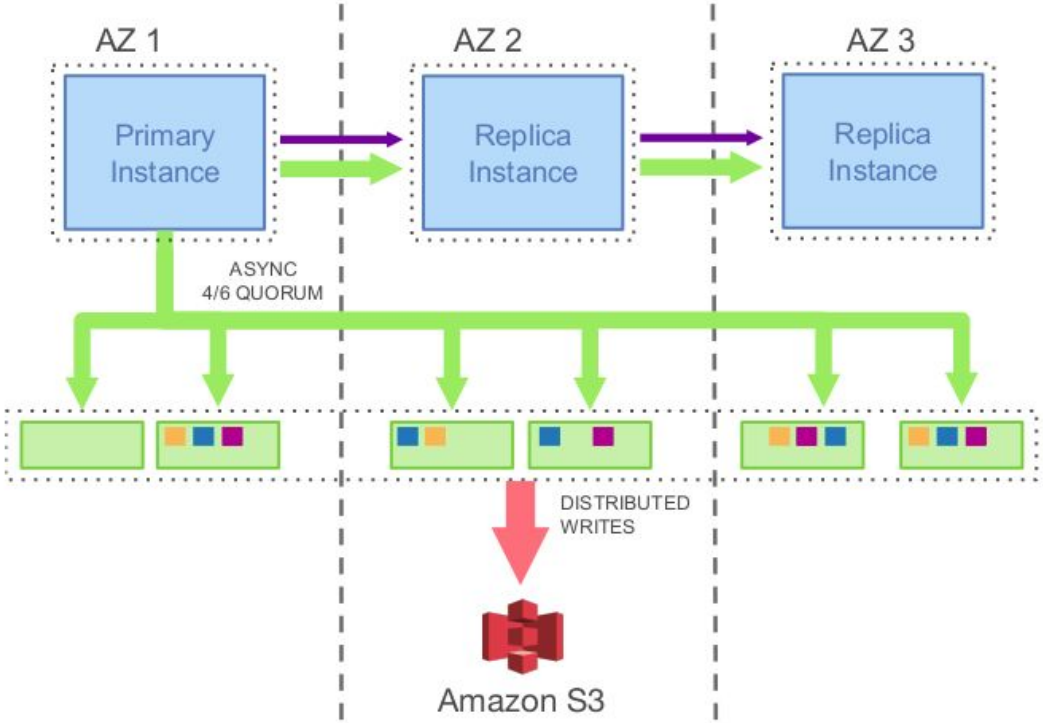
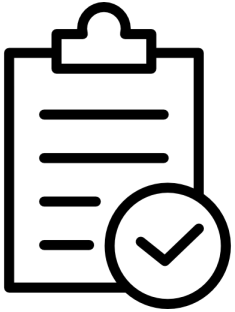


Figure 3: Network IO in Amazon Aurora

Exercises



Constraints: This are theoretical exercises, however you need to submit PRs in Github. You can Create 01.md file and use markdown syntax.

1. What are the good USE CASES for SQL / Relational DBs?
2. What are the good USE CASEs for NoSQL?
3. When you should use Distributed Logs? What use cases?

NoSQL & Distributed Log

Diego Pacheco