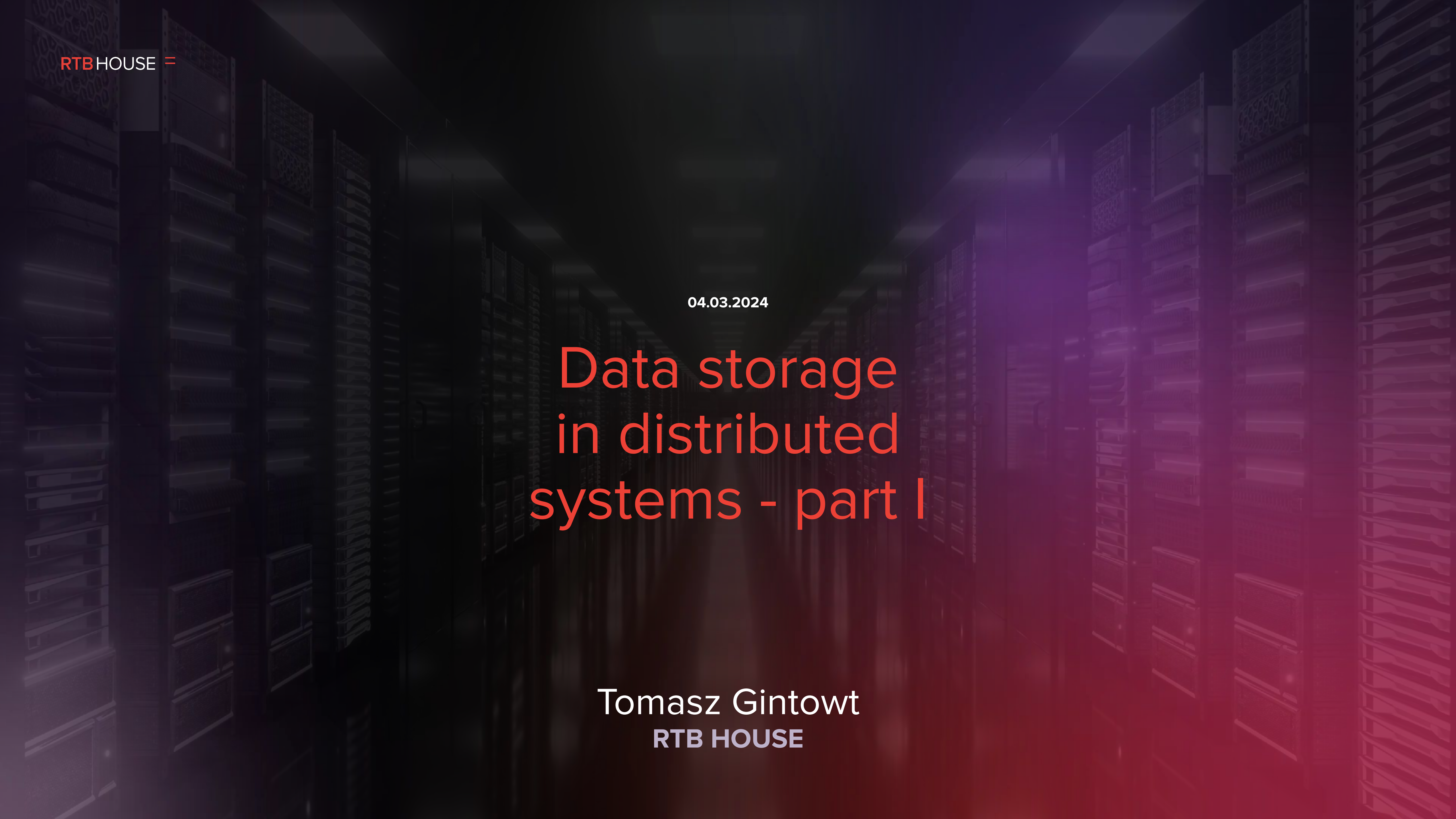


RTB HOUSE

04.03.2024

Data storage in distributed systems - part I

Tomasz Gintowt
RTB HOUSE

Community

DataOps Poland

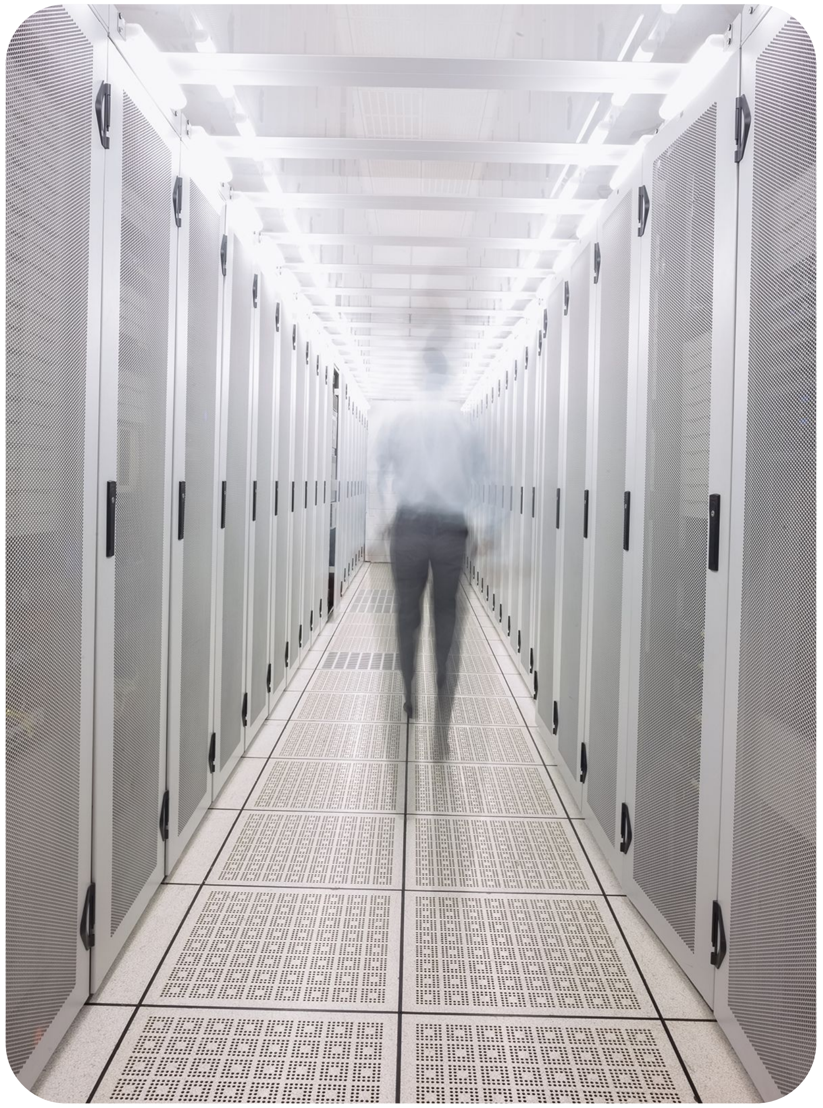
Warsaw PostgreSQL Users Group

What will this lecture be about?

In this part of the course we will consider the topic of data storage technologies for data intensive distributed systems.

We will cover the topics of:

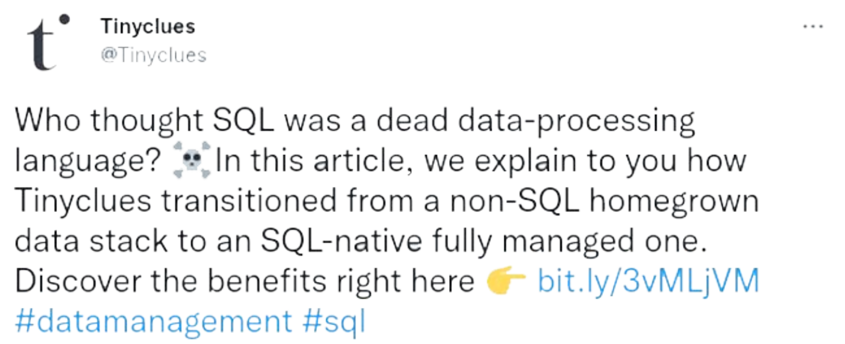
- The aspects of running a traditional RDBMS in a distributed system.
- The common problems with relations databases in distributed environment.



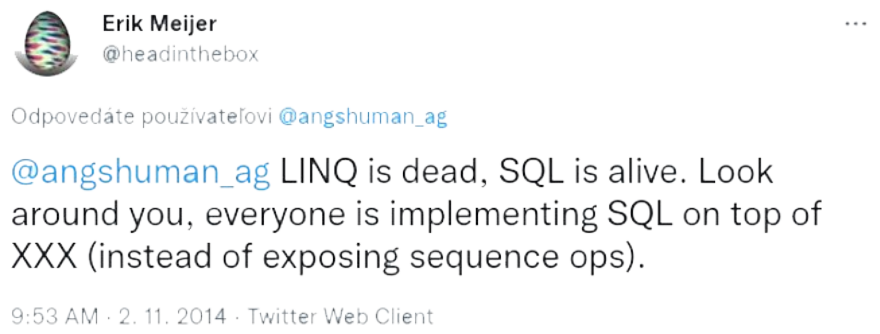
A little bit history

RDBMS	ORACLE	MySQL	Microsoft SQL Server	PostgreSQL
NoSQL	mongoDB	elasticsearch	cassandra	redis
NewSQL	VOLTDB	CockroachDB	Couchbase	
Distributed SQL	yugabyteDB	Cloud Spanner	NUODB	citusdata

A little bit history


Tinyclues
@Tinyclues

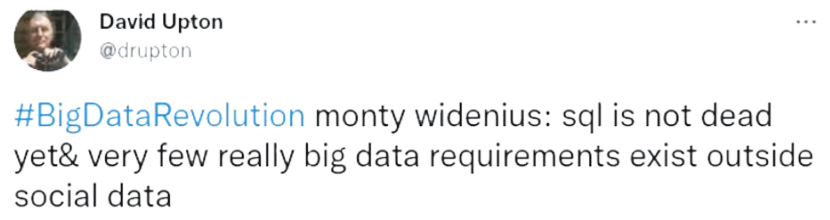
Who thought SQL was a dead data-processing language? ☠️ In this article, we explain to you how Tinyclues transitioned from a non-SQL homegrown data stack to an SQL-native fully managed one. Discover the benefits right here 👉 bit.ly/3vMLjVM #datamanagement #sql


Erik Meijer
@headinthebox

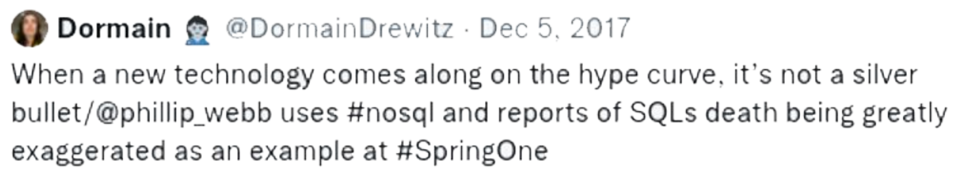
Odpovedáte používateľovi @angshuman_ag

@angshuman_ag LINQ is dead, SQL is alive. Look around you, everyone is implementing SQL on top of XXX (instead of exposing sequence ops).

9:53 AM · 2. 11. 2014 · Twitter Web Client


David Upton
@drupton

#BigDataRevolution monty widenius: sql is not dead yet& very few really big data requirements exist outside social data


Dormain 🧟 @DormainDrewitz · Dec 5, 2017
When a new technology comes along on the hype curve, it's not a silver bullet/@phillip_webb uses #nosql and reports of SQLs death being greatly exaggerated as an example at #SpringOne

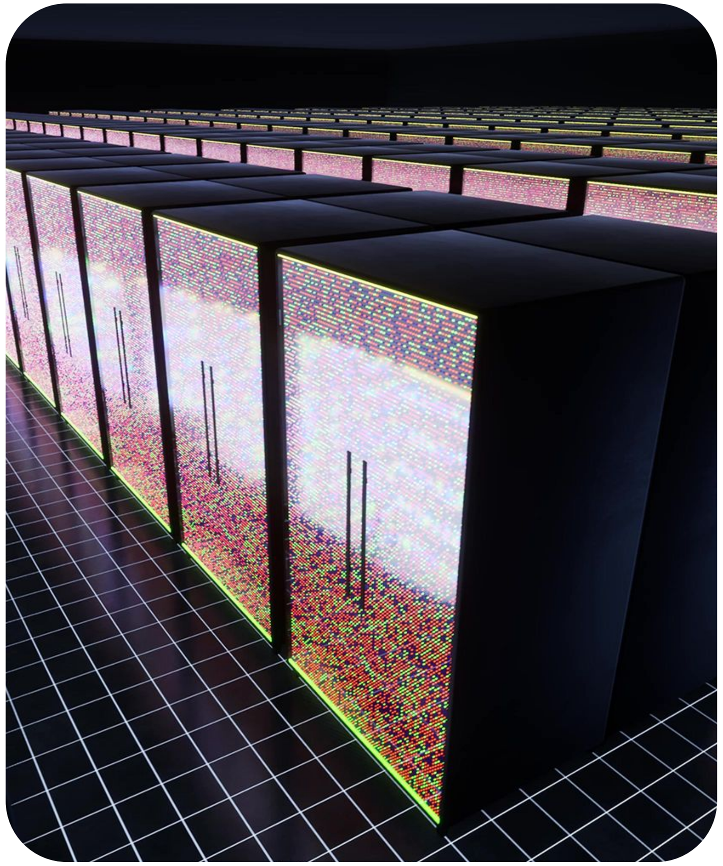
Distributed Storage

Relational Database
Management Systems

RDBMS

Why still use RDBMS?

- Well established flexible data model.
- Stable and battle-proven.
- Powerful Structured Query Language.
- Full support for ACID transactions.
- Swiss army knife - supports most corner cases.
- Well documented.



RTB HOUSE

Transactions

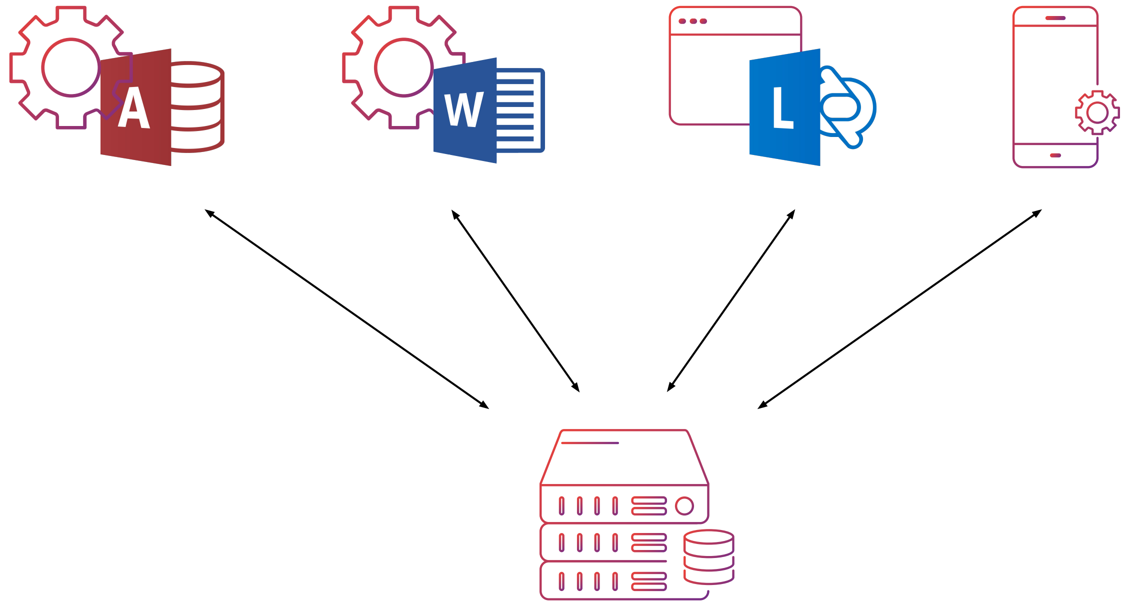
The main advantage of using single node relational databases is the fact that they fully support the concept of a transaction.

- Atomicity - operations within transaction either completely succeed or abort.
- Consistency - operations within transaction do not violate invariants or data integrity.
- Isolation - different transactions do not interfere with each other.
- Durability - the data has been written to disk (including WAL).

For a distributed system the isolation part is the most interesting one.

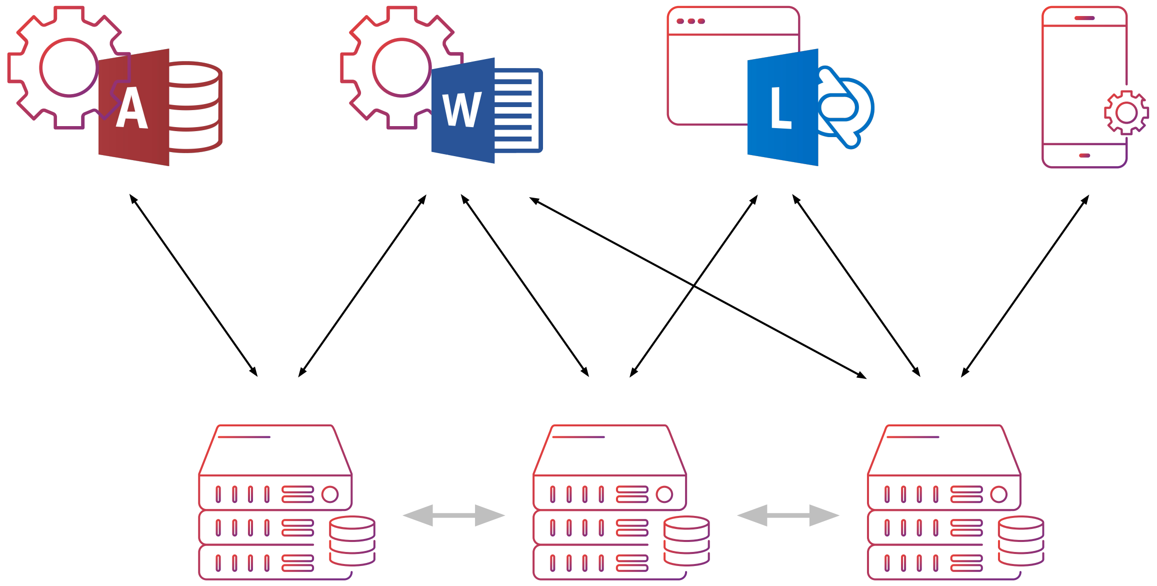
RDBMS as a silo

- Single server
- Very high cost
- 128 CPU, 512 G RAM
- Long mean time between failures
- Long mean time to repair
- SPOF - single point of failure



RDBMS as a cluster

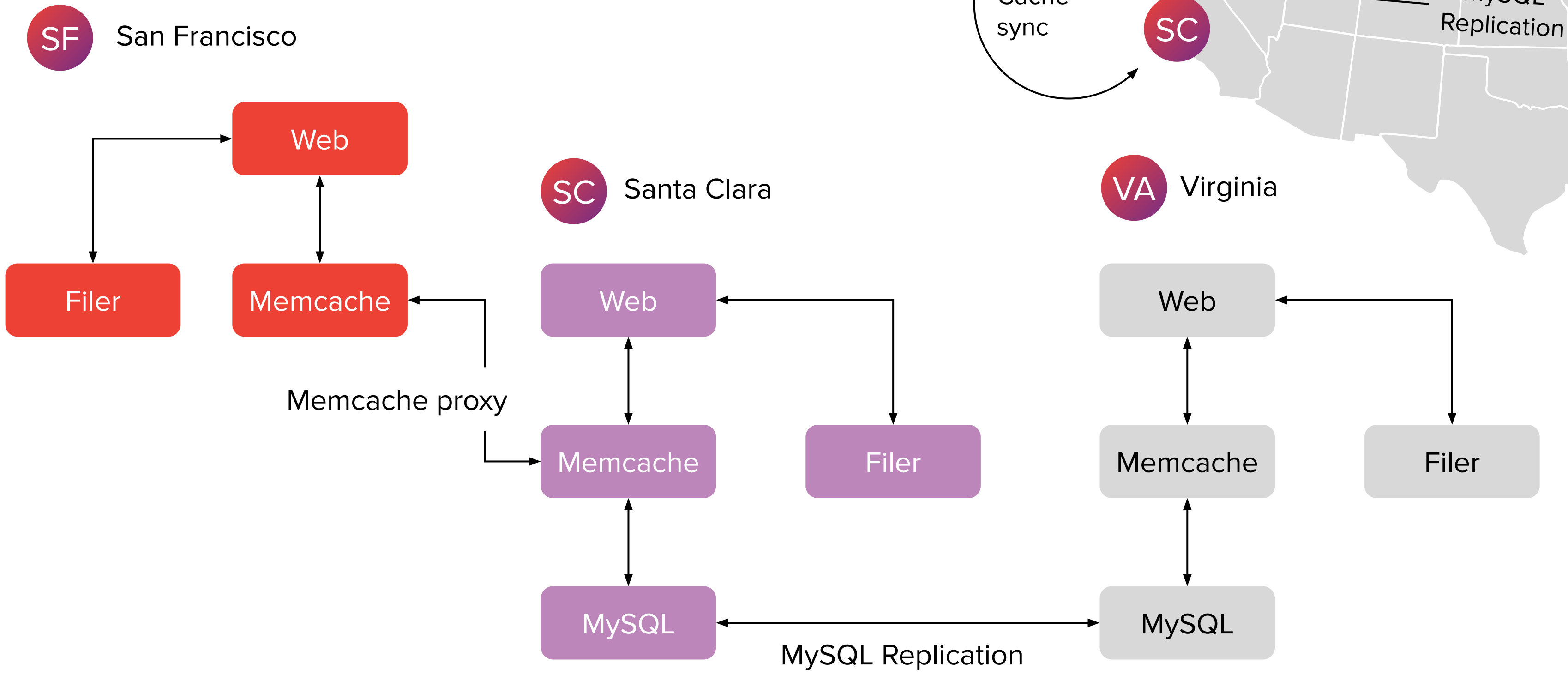
- Multiple servers
- Reasonable cost
- 3 x (64 CPU, 128 G RAM)
- Short mean time between failures
- Short mean time to repair
- High availability

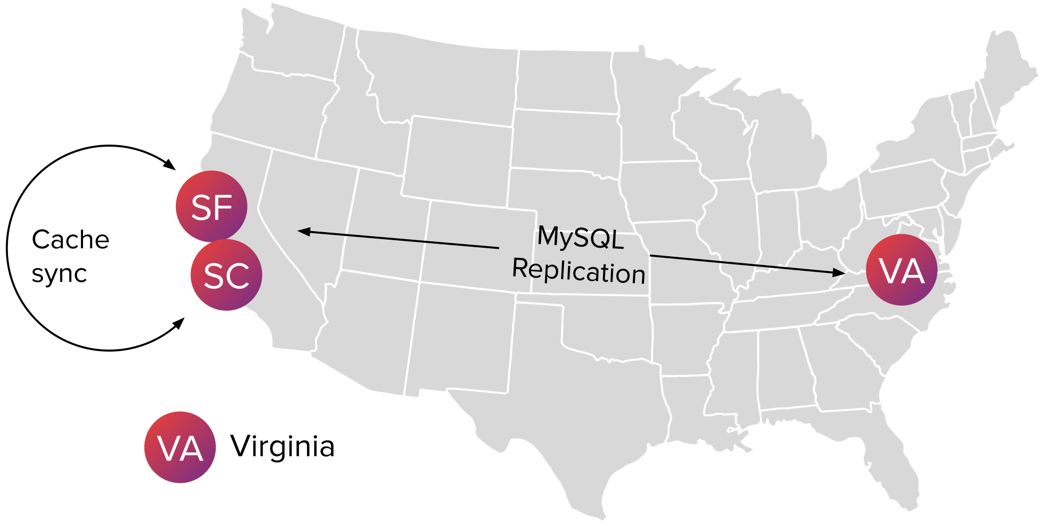


RDBMS as a cluster

- Multiple regions
- Master-Slave replication
- Master-Master replication
- Low latency
- Millions of request per second
- High availability

Facebook database - MySQL





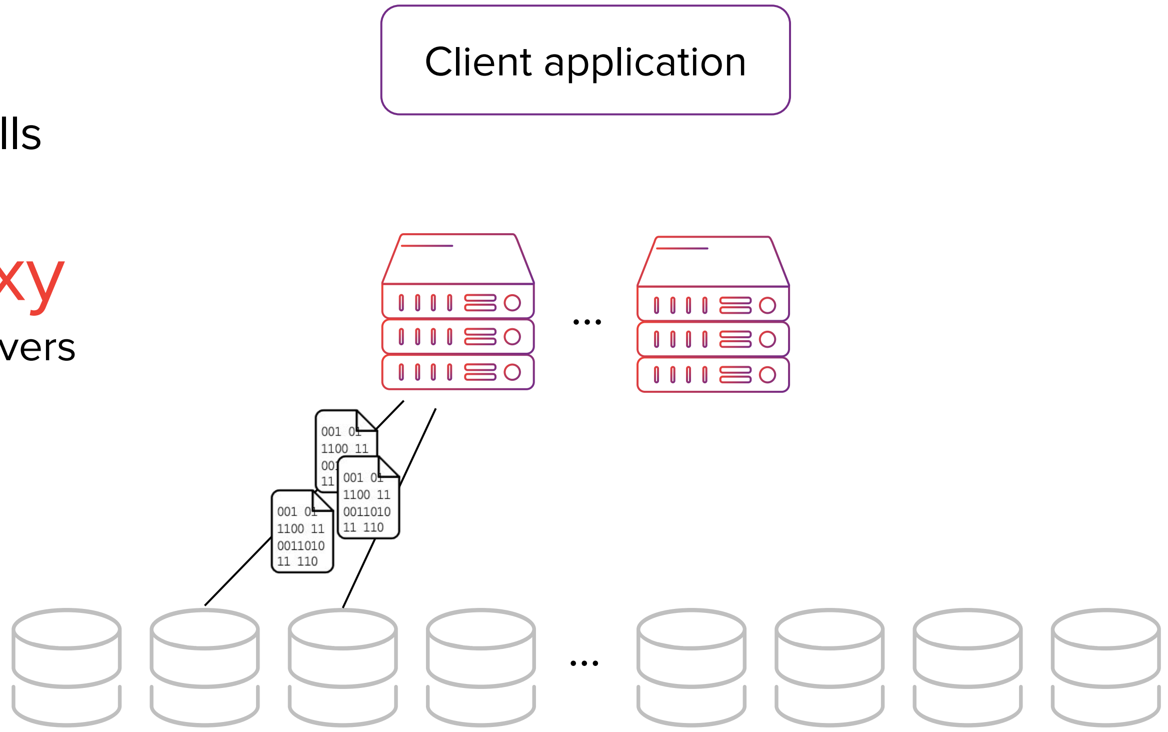
Skype database - PostgreSQL

Procedure calls

PL / Proxy

partitioning servers

Database partitions

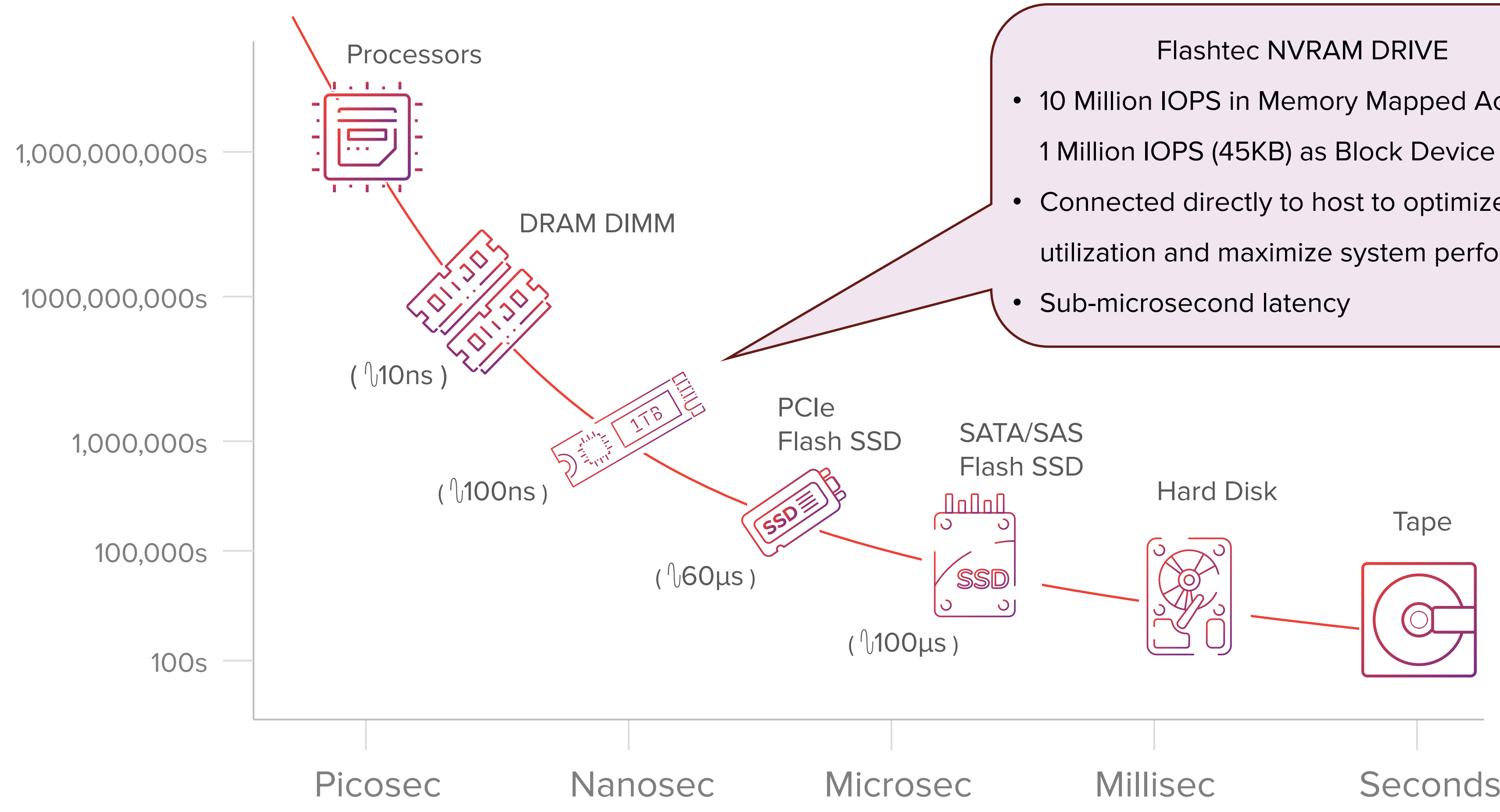


Distributed Storage

Hardware

RTB HOUSE =

How fast is your disk ?



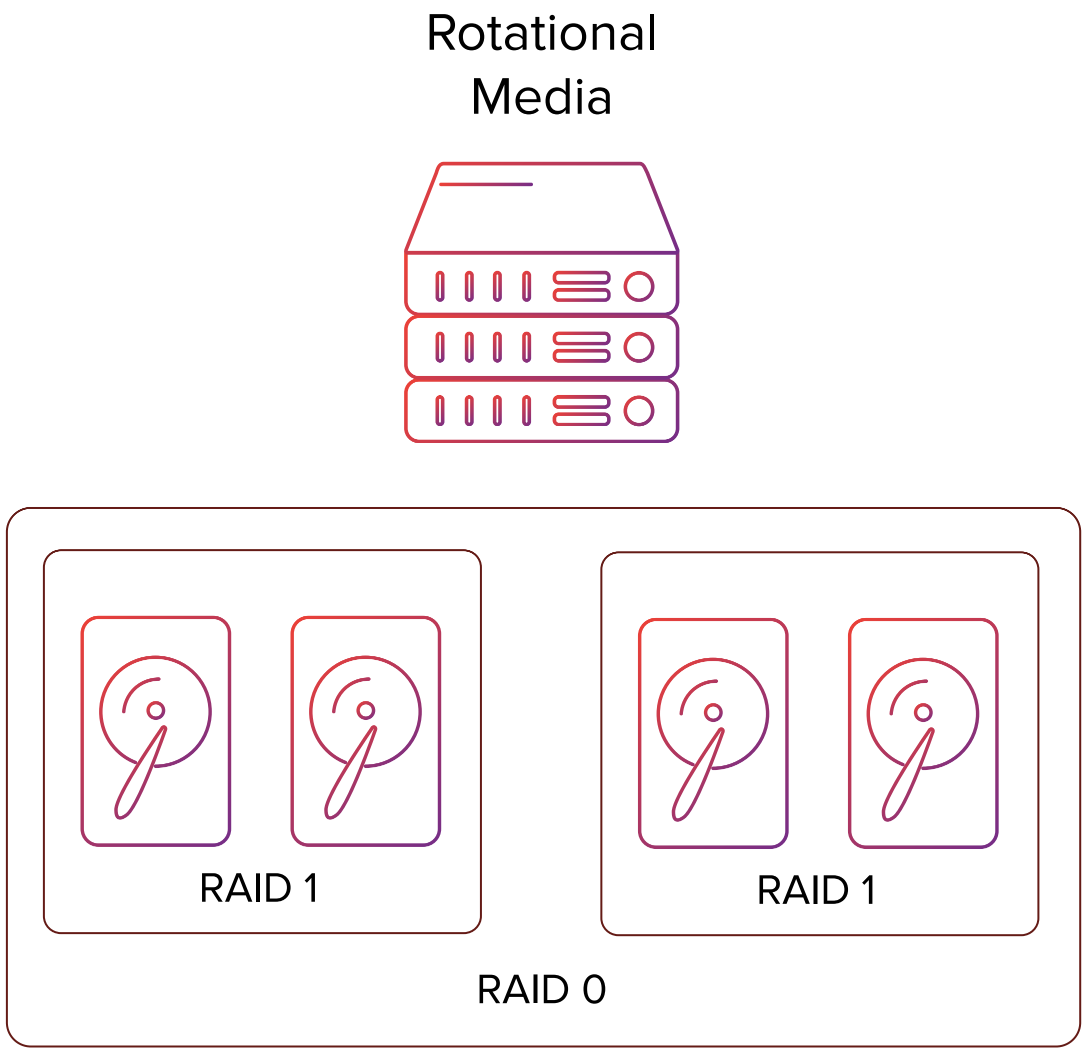
Flashtec NVRAM DRIVE

- 10 Million IOPS in Memory Mapped Access, 1 Million IOPS (45KB) as Block Device
- Connected directly to host to optimize CPU utilization and maximize system performance
- Sub-microsecond latency

RTB HOUSE

Hardware Architecture for RDBMS

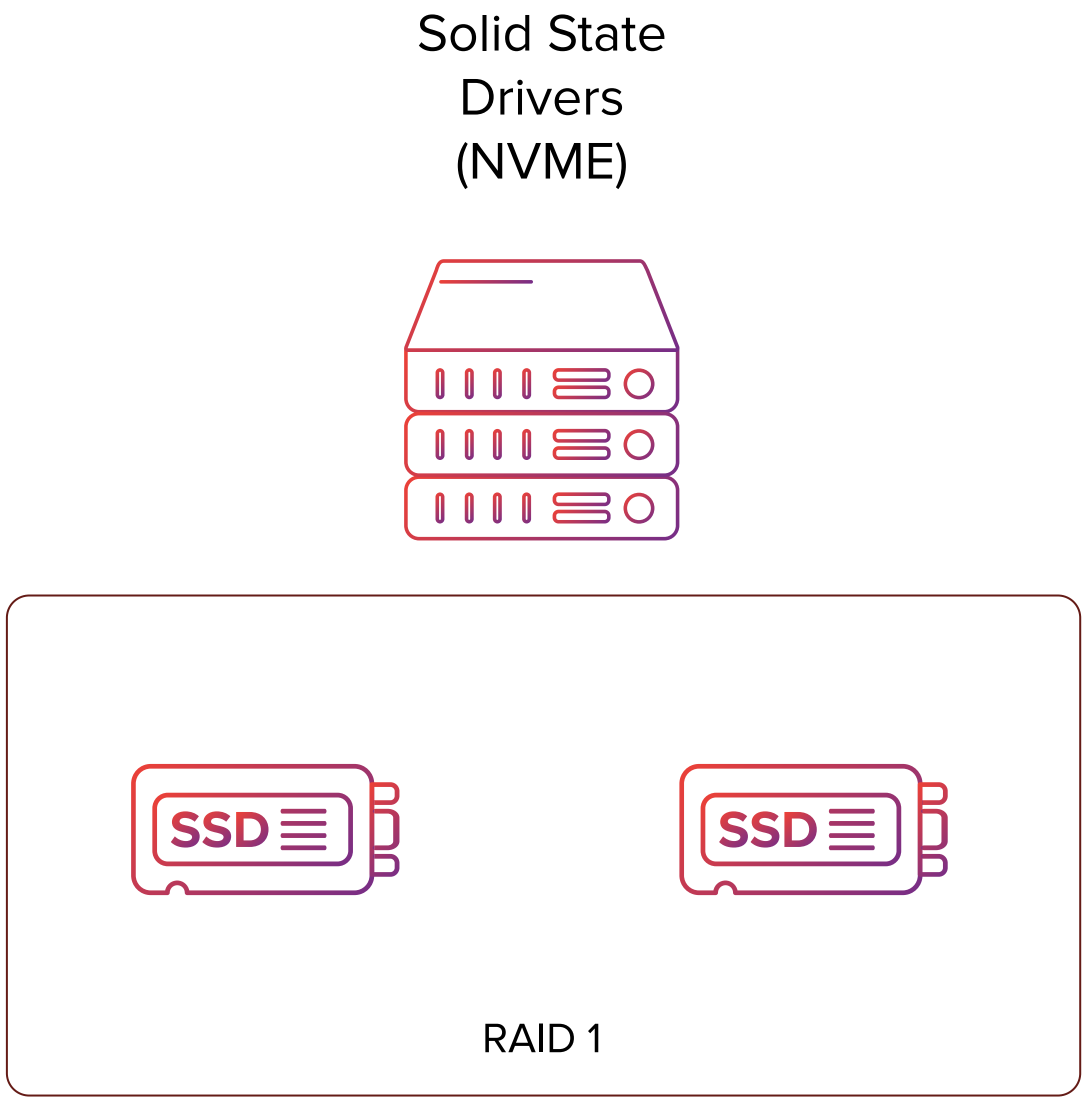
- With rotational media in DAS (Direct Access Storage) configuration the latency of the storage device is the performance bottleneck in RDBMS applications. To mitigate use the RAID 10 (1 + 0) configuration.
- In this configuration the striped data is additionally mirrored for better fault tolerance.
- RAID capabilities can be either provided by system software or dedicated hardware controllers.



RTB HOUSE

Hardware Architecture for RDBMS

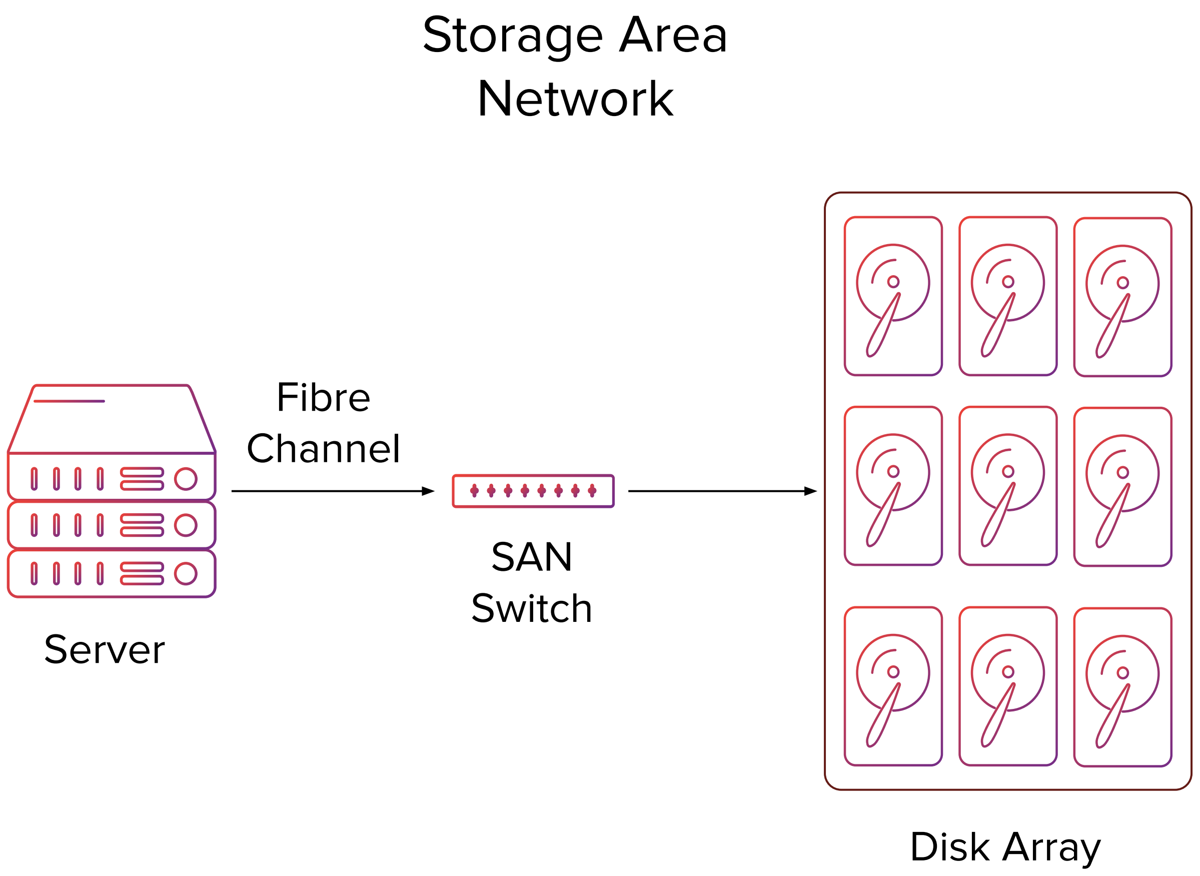
- With solid state media in DAS (Direct Access Storage) configuration, the devices connected directly to the PCI Express bus are no longer the performance bottleneck.
- In this scenario the usual configuration is to use RAID 1 for drive redundancy.



RTB HOUSE =

Hardware Architecture for RDBMS

- The data storage for database can be also accessed via SAN.
- Not as efficient as directly accessed storage.
- Many vendors offer custom proprietary solutions.
- Offers more flexibility regarding storage management.

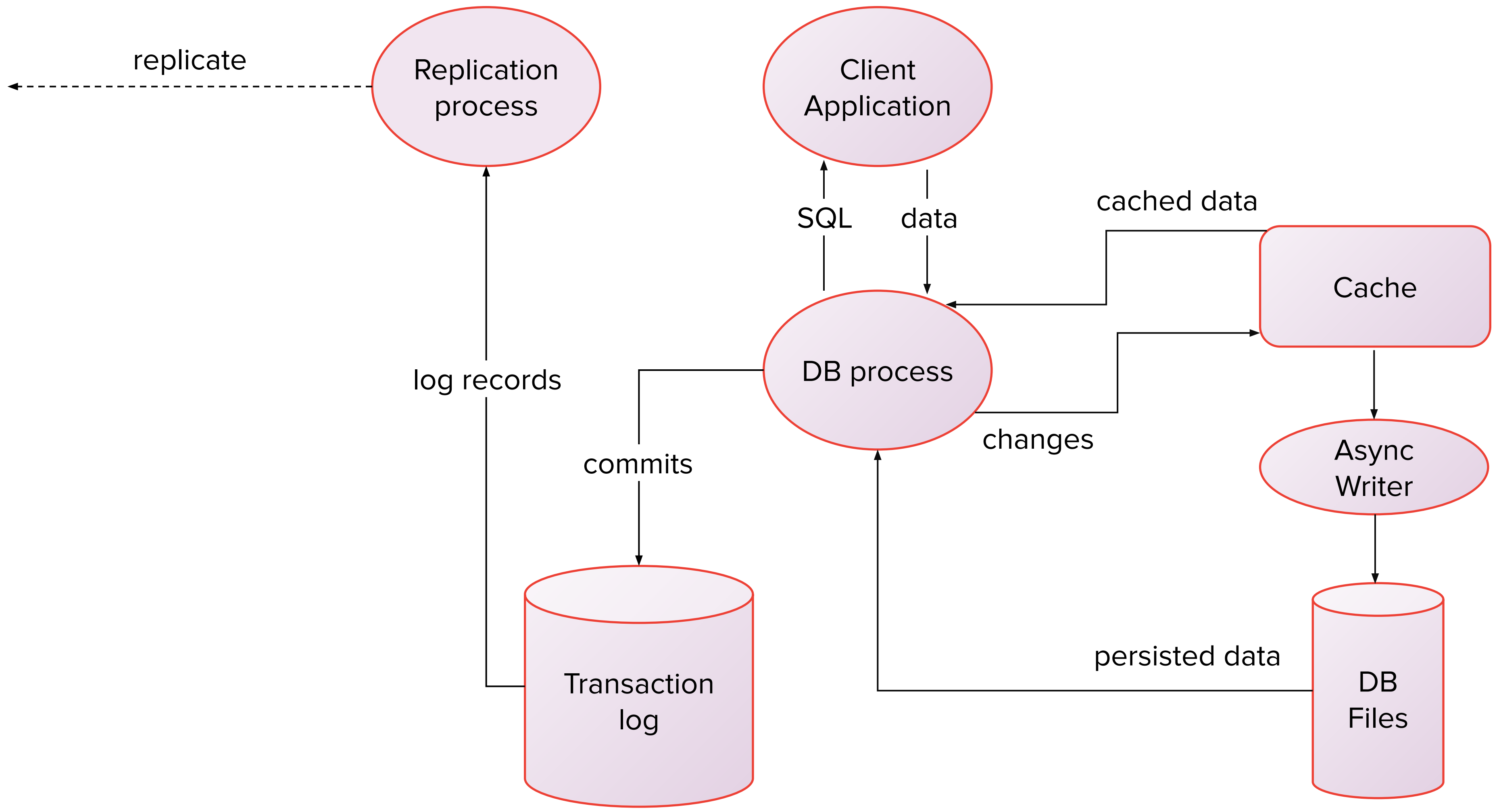


Distributed Storage

Replication

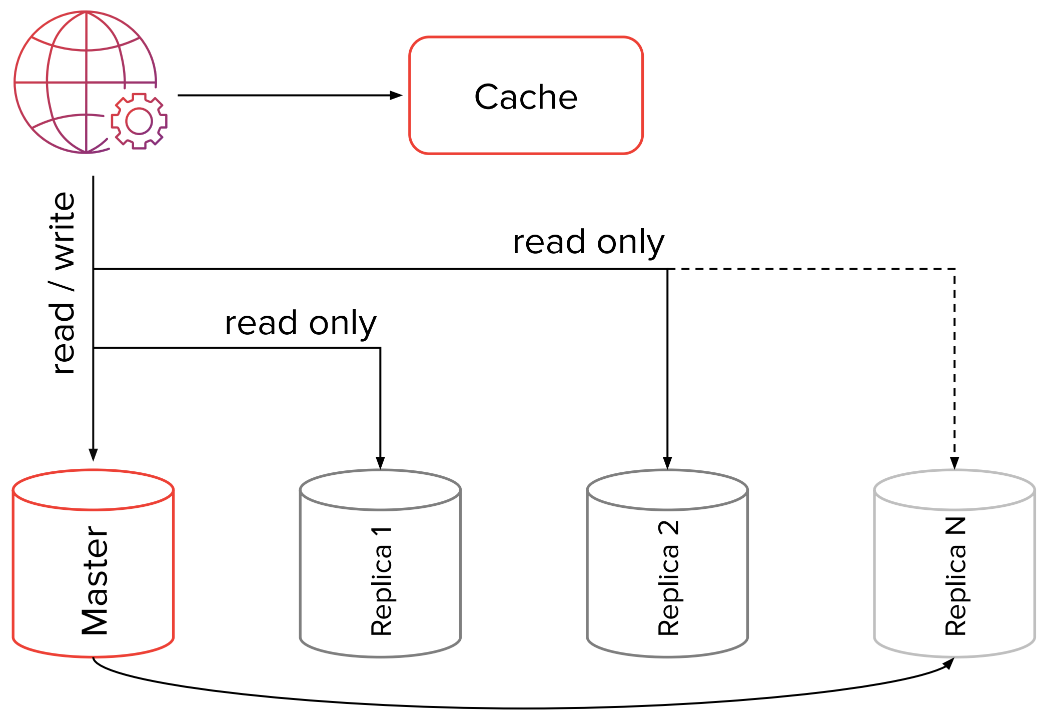
RTB HOUSE =

Typical RDBMS Process



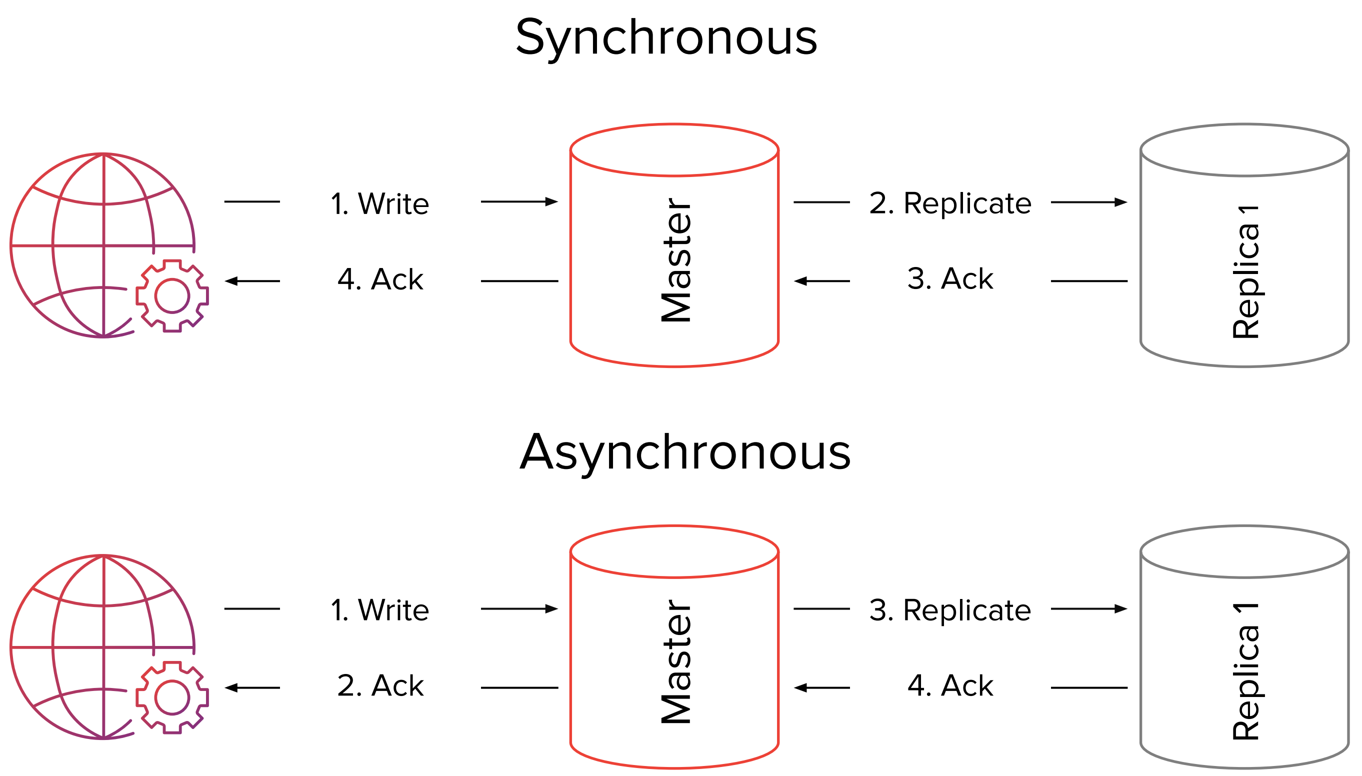
RTB HOUSE

Typical Architecture Involving RDBMS



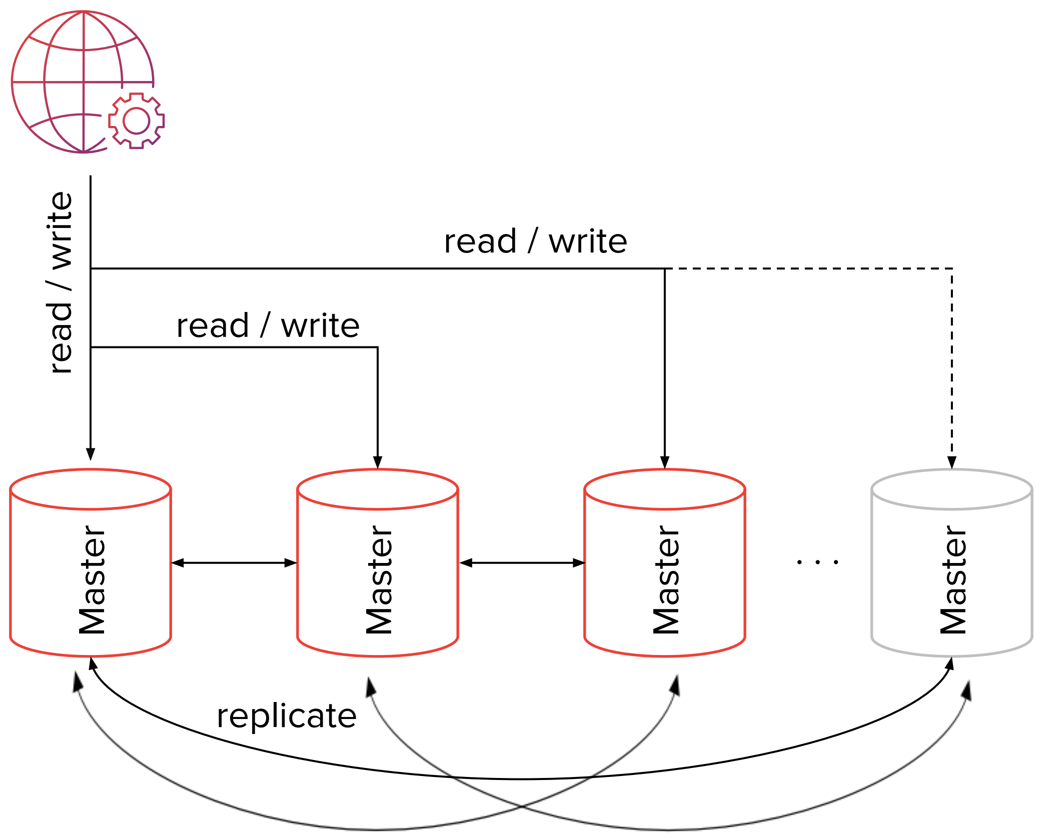
RTB HOUSE =

RDBMS Replication



RTB HOUSE

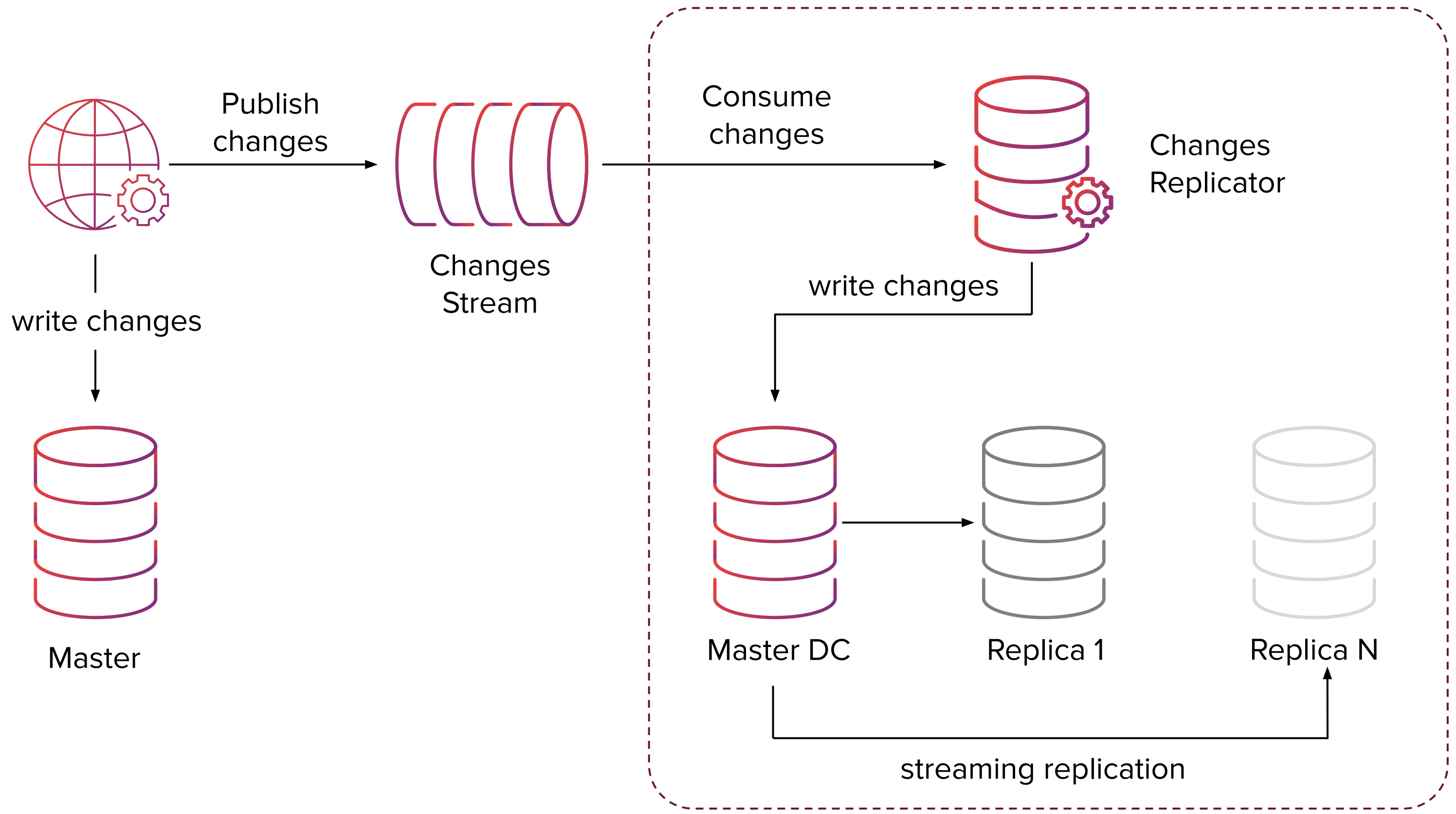
Multi-Master Replication



- Adds complexity.
- Unsupported natively by RDBMSes.
- Inevitable conflicts with non-trivial methods of resolution

RTB HOUSE

Custom Cross-DC Replication - Architecture



RTB HOUSE =

RDBMS Replication

An (incomplete) selection of possible methods:

- Log Shipping - copying completed WAL segments, better resource utilisation, higher risk of reduced durability (lost changes).
- Log Streaming - sending individual WAL entries over an open connection, higher resource utilisation, lesser risk of reduced durability.
- SQL replication - possible inconsistent results between replica nodes (ie NOW() function yields different results).
- Logical replication - changes are published in a defined message format and are applied by subscriber nodes.
- File System (Block Device) Replication - all changes to a file system are mirrored to a file system residing on another computer.

Replication Burden

Replication adds additional burden to primary node with growing number of follower nodes.

Cascading replication - follower nodes cascade changes to other followers.

Custom solution - when replicating between DC's:

- Allow only aggregates in database - denormalization with JSON columns.
- Force applications that mutate data to post aggregated changes on queue system.
- Feed "followers" on target DC's from queues (Kafka).
- Run checkers to fix consistency issues .
- The performance of replication lag consumption is scalable (more change consumers)

Distributed Storage

Challenges

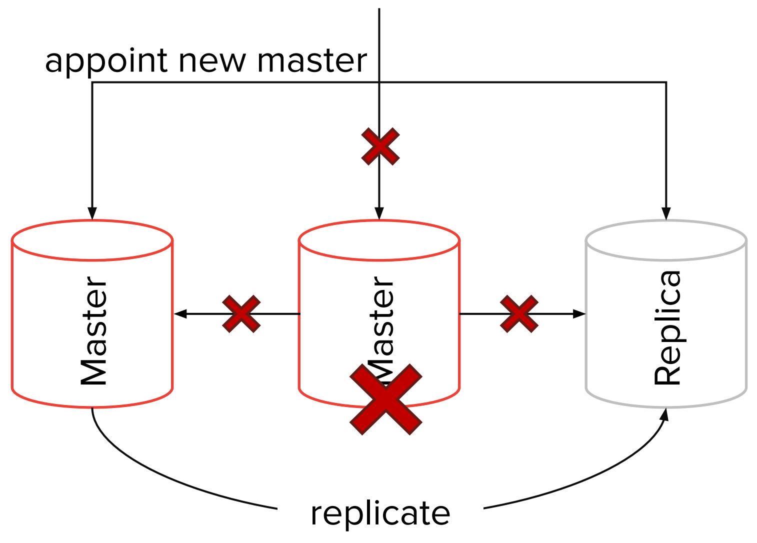
RTB HOUSE =

Where is Master?

RTB HOUSE

Where is Master ?

- Additional layer PROXY or
- Load balancer with Virtual IP.
- Service Discovery with DNS.
- Dedicated tool like ProxySQL or PgPool-II.
- All DML queries should go to Master.



RTB HOUSE =

Connection pooling

Load Balancing and Connection Pooling

SELECT queries on RDBMS may be distributed among several replicas for load balancing reasons, whereas write queries are usually distributed to the master node.

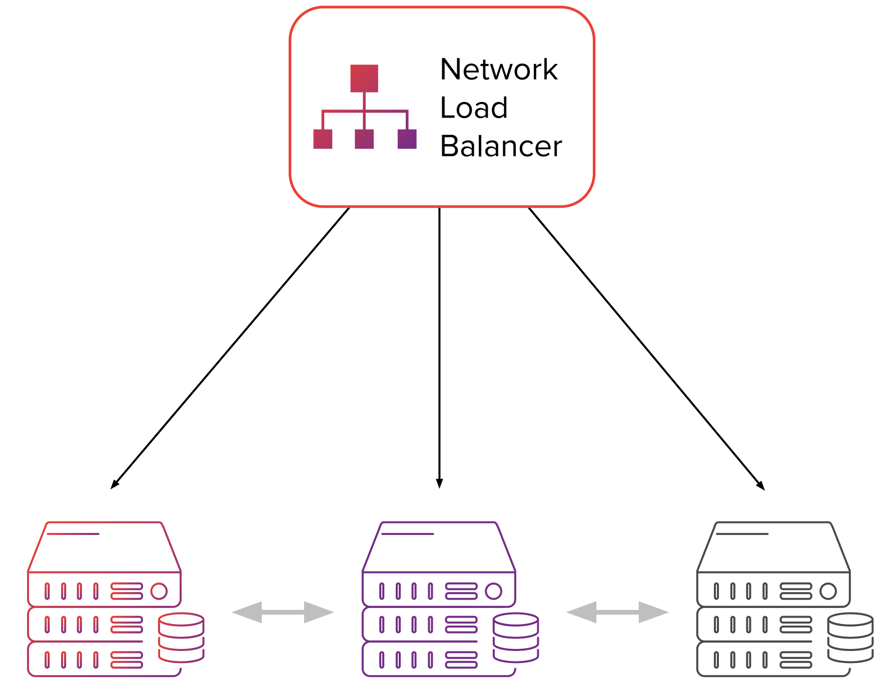
Establishing connections to a database can be expensive so it is a good idea to set a pool of connections and reuse them.

Both load balancing and connection pooling can be achieved in two ways:

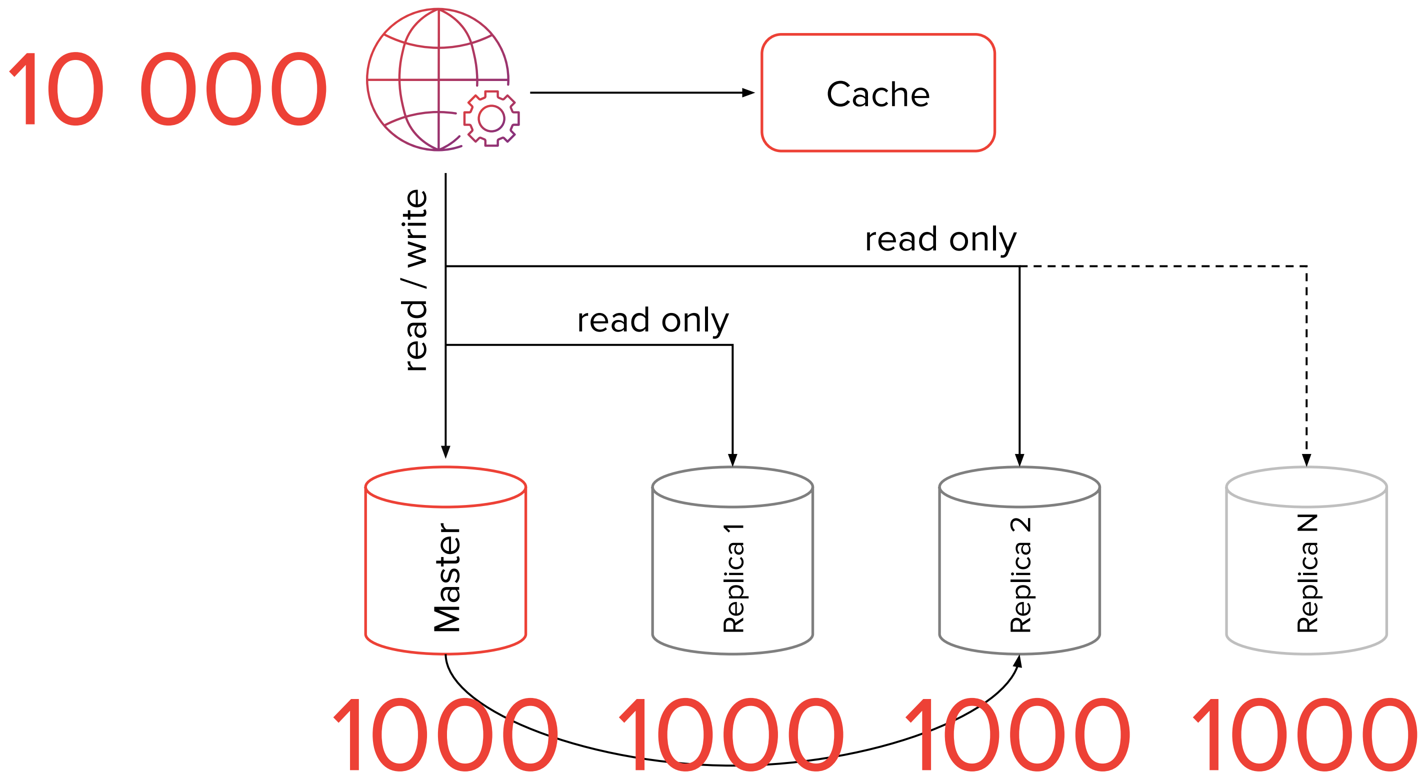
- Internally by an instance of the application.
- By a centralized component.

RTB HOUSE =

Load Balancing



Connection Pooling



RTB HOUSE =

High Availability

RDBMS Failover

When the primary database fails and needs to be replaced by one of the replica nodes.

MANUAL

the new master has to be promoted manually by system's operator. This will usually mean partial unavailability for a RDBMS.

AUTOMATIC

the database cluster manager will automatically promote new master from the set of available replicas

RTB HOUSE

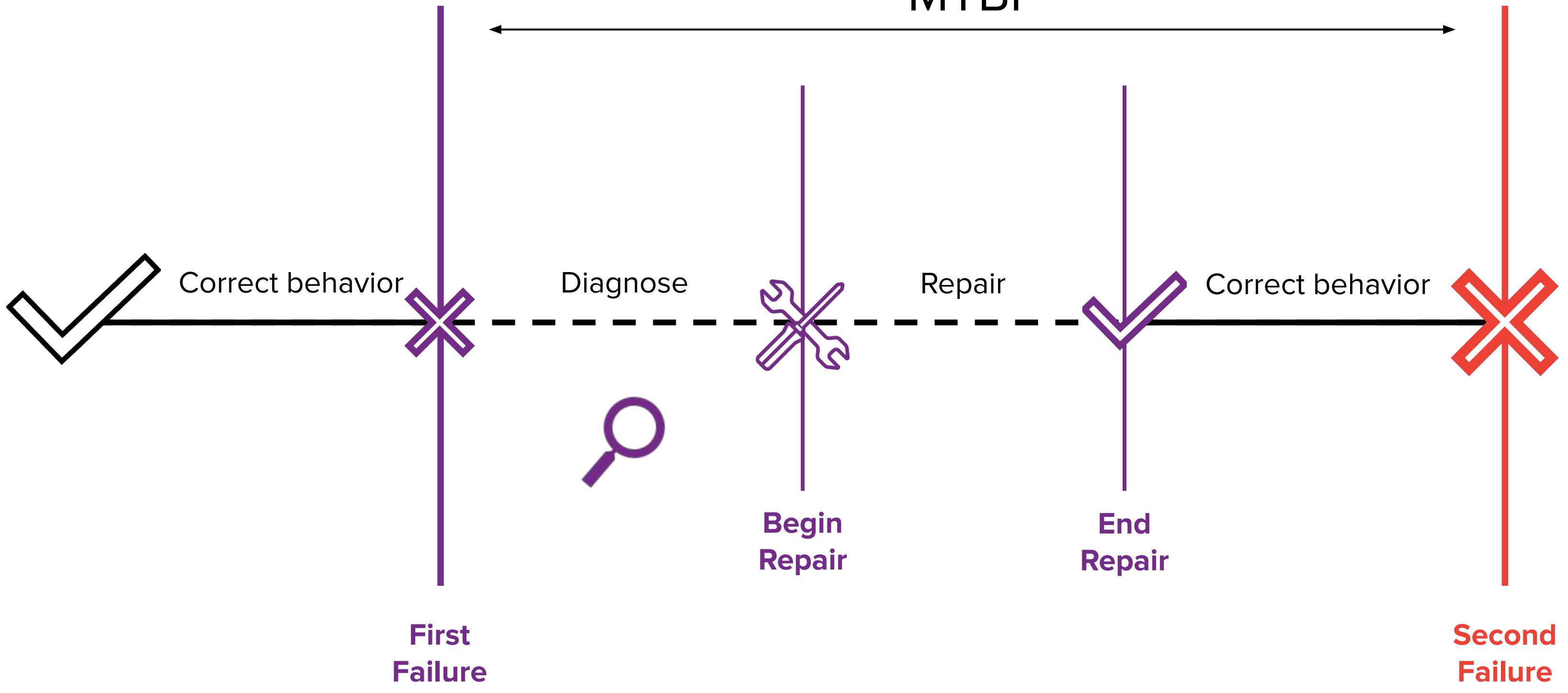
RDBMS High Availability

Failure Is Just Part of the Game

MTTR
Mean Time
To Repair

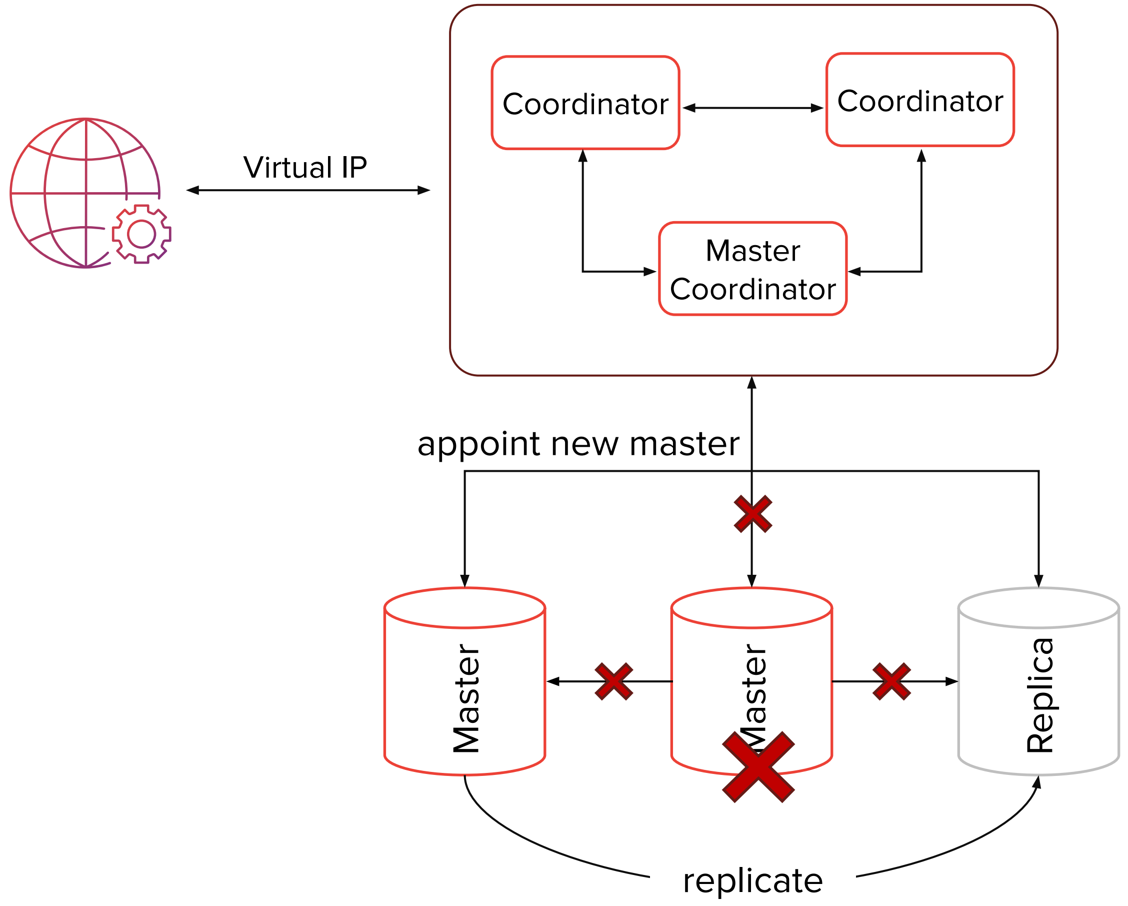
MTBF
Mean Time
Between Failures

RDBMS High Availability



RTB HOUSE

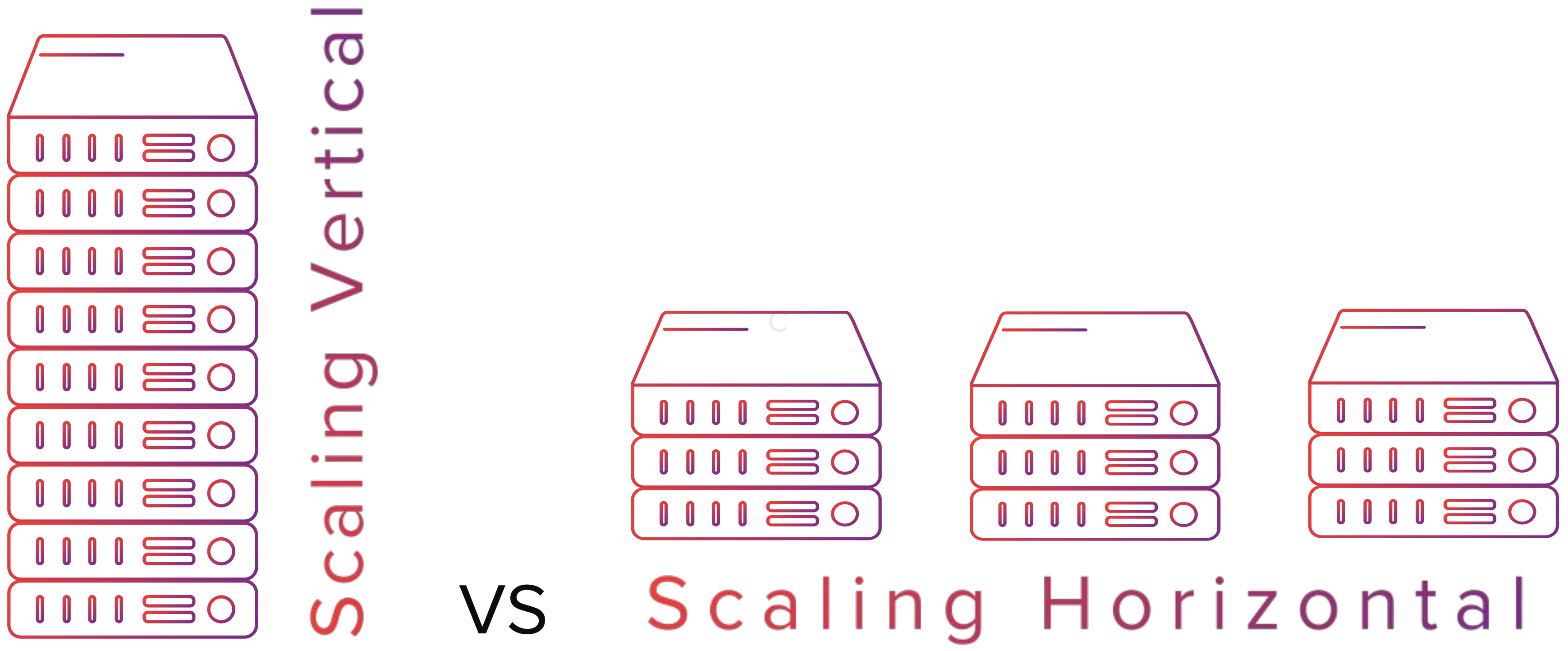
RDBMS Failover



Scaling

RTB HOUSE

RDBMS Scaling



RTB HOUSE =

RDBMS Vertical Scaling

How to scale the RDBMS vertically with losing as little availability as possible?

- CPU, RAM - these are hot swappable elements in the high-end server equipment.
- Increasing disk capacity - in some SAN solutions this can be done with zero downtime by adding/replacing disks.
- On commodity hardware it usually requires short downtime or partial unavailability (read only mode).

RTB HOUSE

RDBMS Vertical Scaling

In case of commodity servers the exemplary strategies to scale capacity can be used:

By using Logical Volumes:

1. Add new disk to the system and expand the current logical volume and resize the filesystem online.
2. Replace the existing logical volume:

- Add new disk and create new logical volume.
- Synchronize the data between the old and new logical volumes.
- Disable database for a short moment and replace the volumes.
- Re-enable the database.

RTB HOUSE =

RDBMS Vertical Scaling

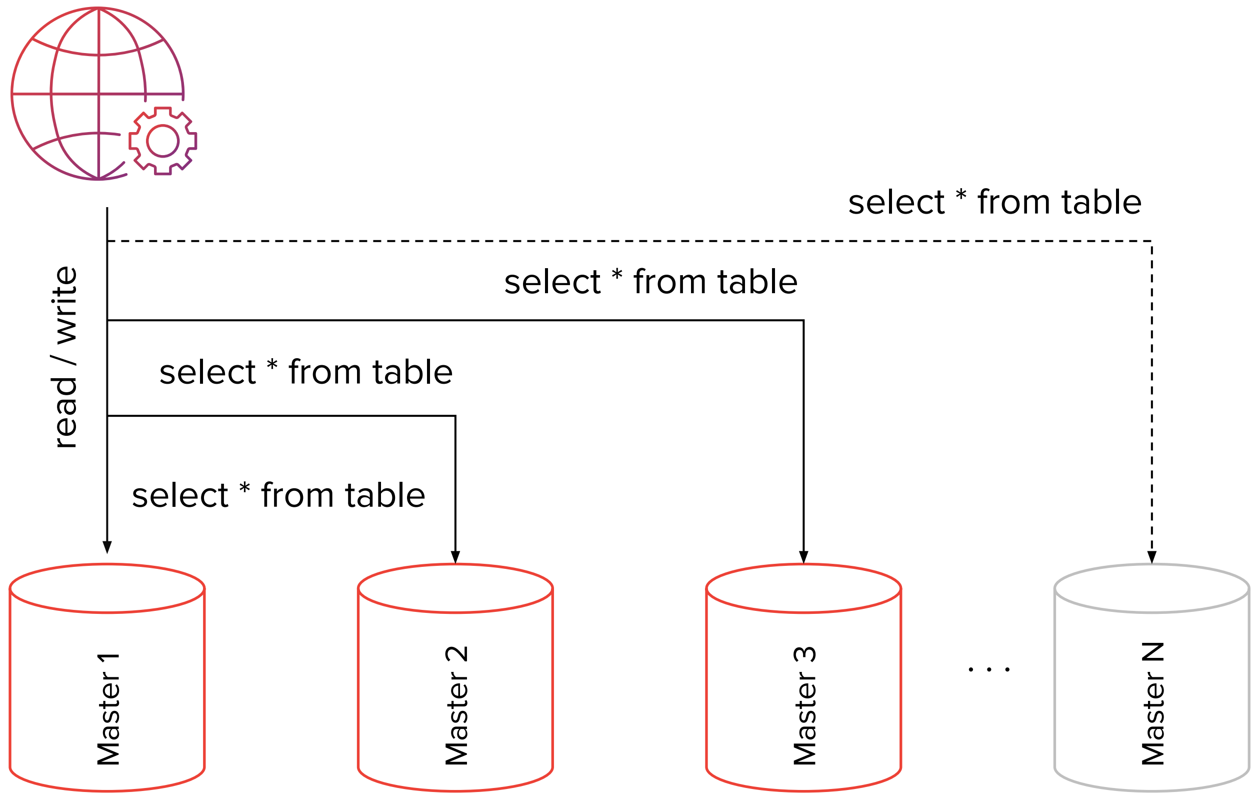
In case of commodity servers the exemplary strategies to scale capacity can be used:

By adding a new replica:

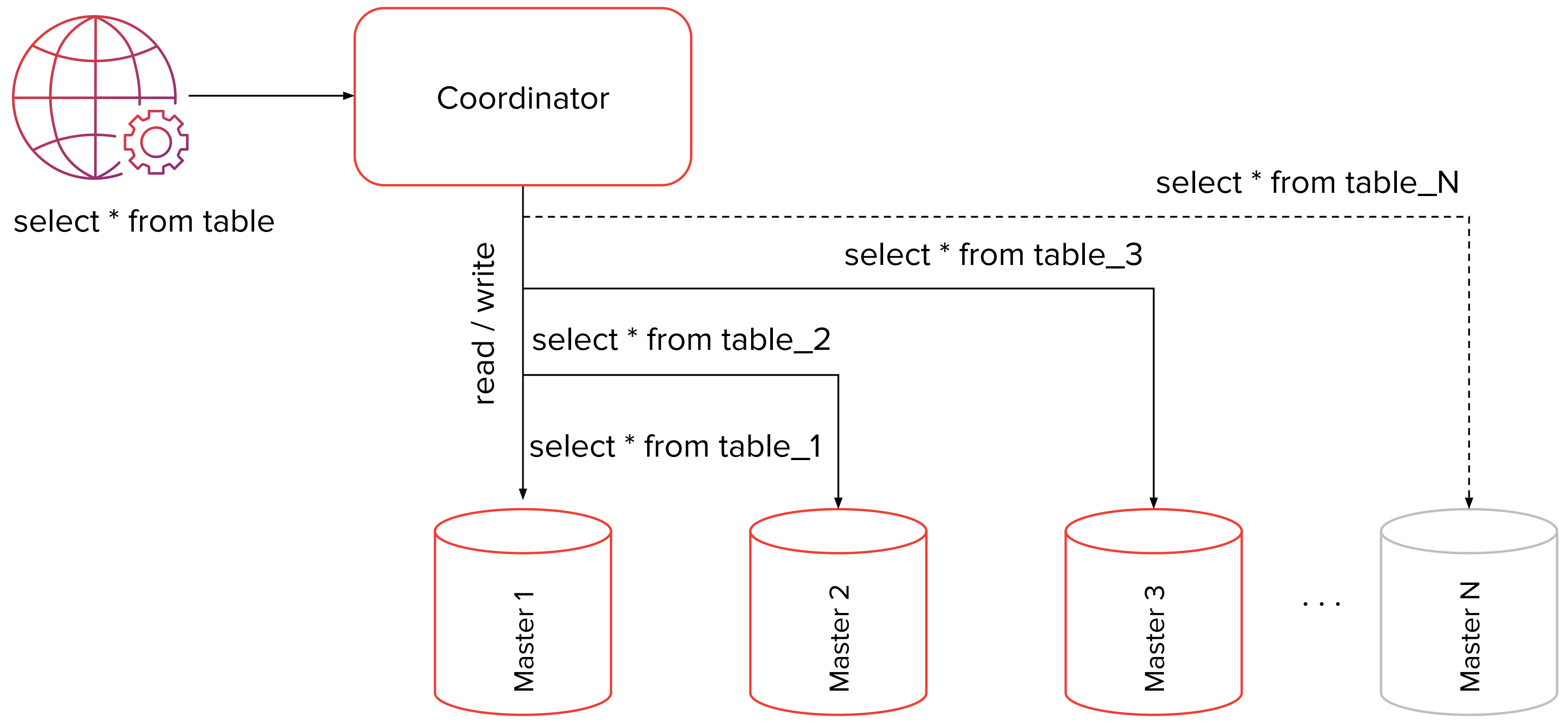
- Add new machine with desired new capacity (and other specification).
- Configure it as a replica and let it synchronize with the master.
- Wait for the replica to fully synchronize and promote it to leadership.

RTB HOUSE

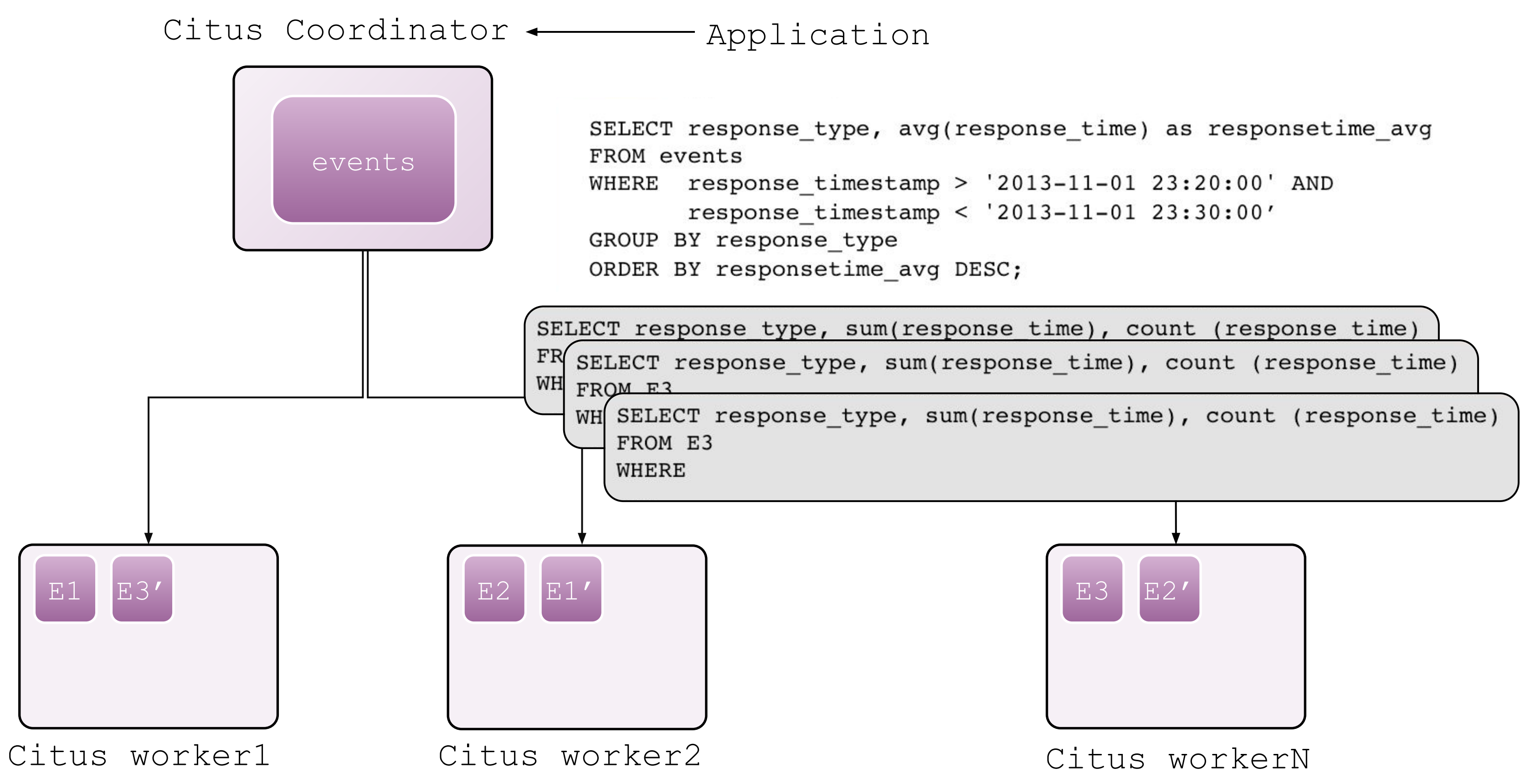
RDBMS Horizontal Scaling (Sharding)



RDBMS Horizontal Scaling (Sharding)



RDBMS Horizontal Scaling (Sharding)



```
SELECT response_type, avg(response_time) as responsetime_avg
FROM events
WHERE  response_timestamp > '2013-11-01 23:20:00' AND
       response_timestamp < '2013-11-01 23:30:00’
GROUP BY response_type
ORDER BY responsetime_avg DESC;
```

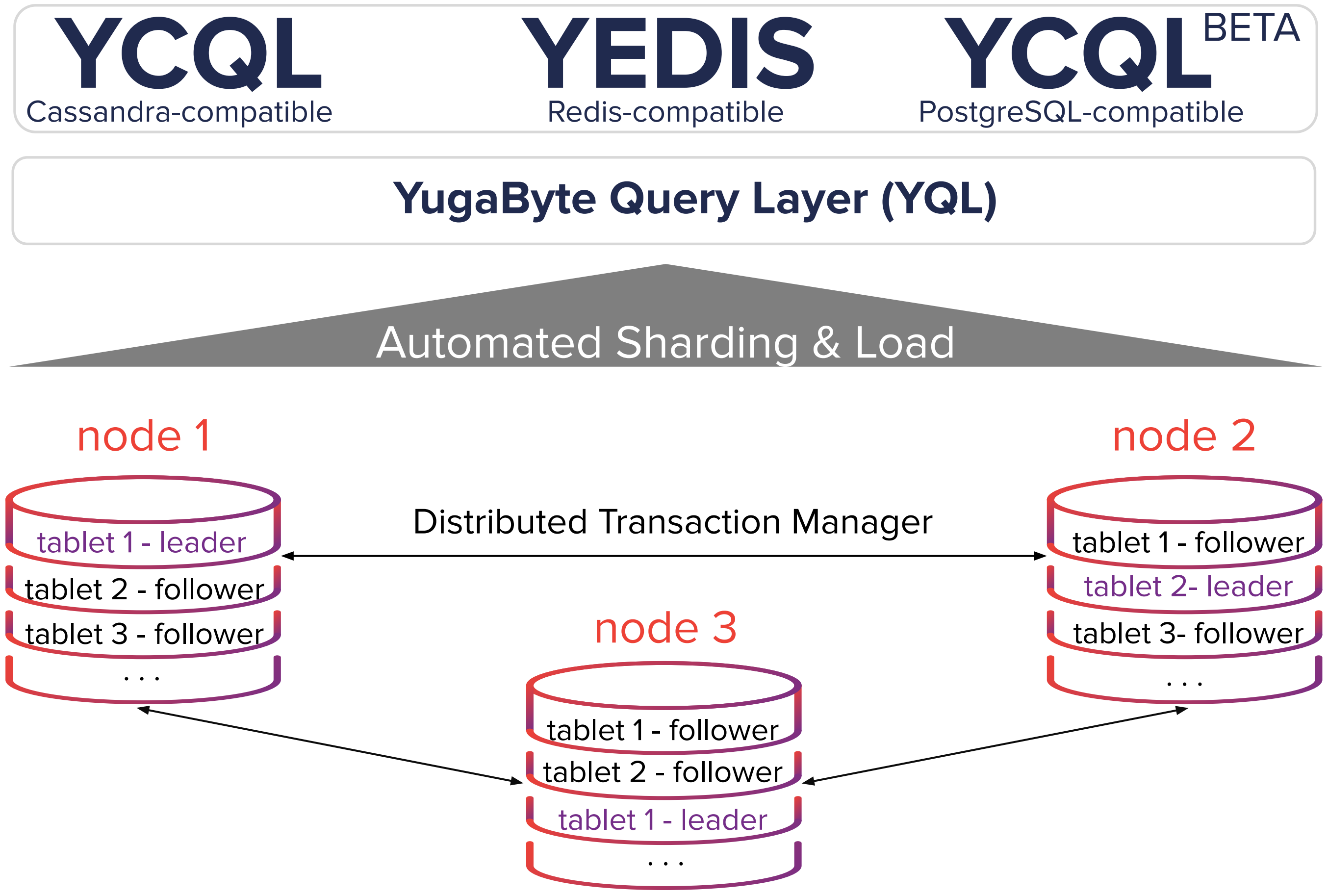
```
SELECT response_type, sum(response_time), count (response_time)
FR
WH
```

```
SELECT response_type, sum(response_time), count (response_time)
FROM E3
WH
```

```
SELECT response_type, sum(response_time), count (response_time)
FROM E3
WHERE
```

https://docs.citusdata.com/en/v7.0/aboutcitus/introduction_to_citus.html

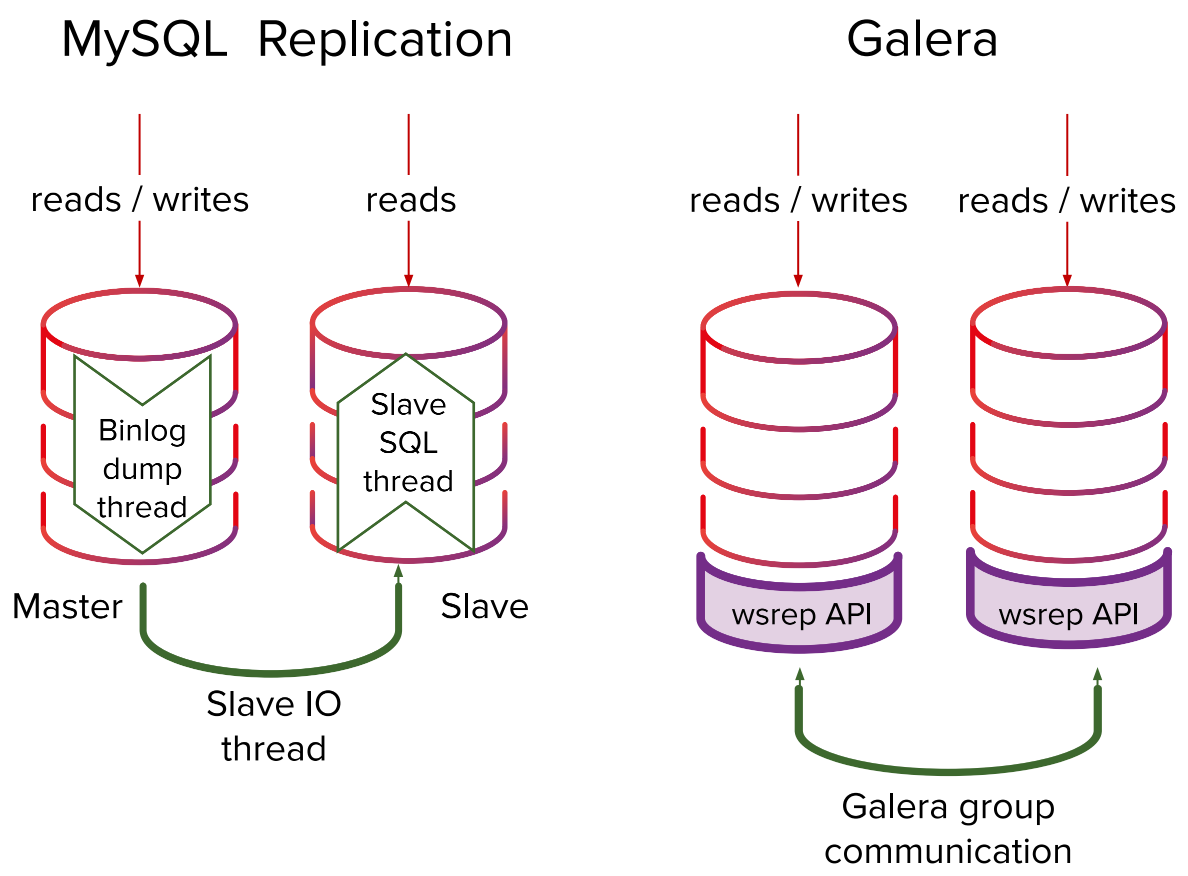
RDBMS Horizontal Scaling



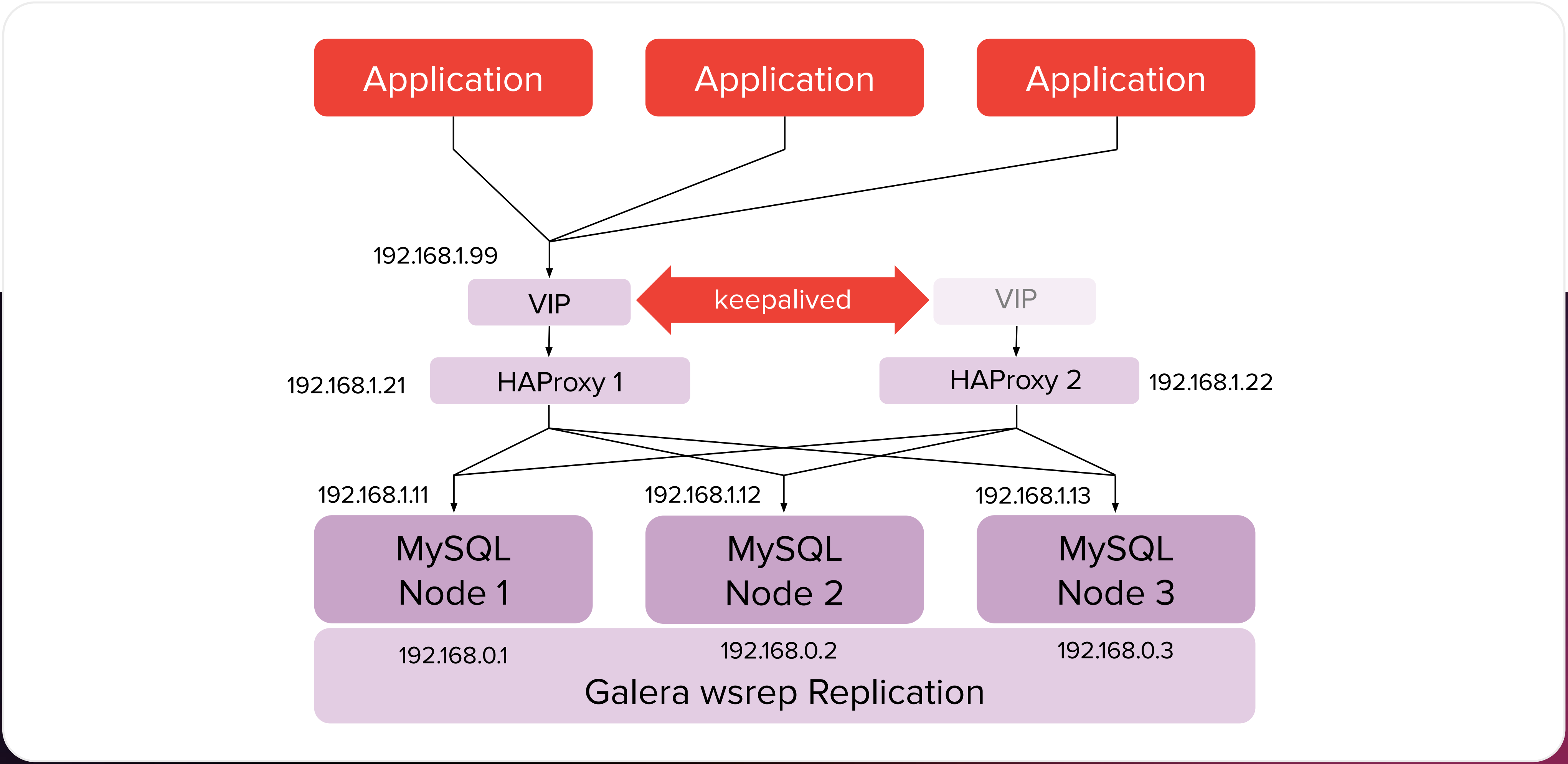
https://www.yugabyte.com/blog/ysql-architecture-implementing-distributed-postgresql-in-yugabyte-db/

RTB HOUSE

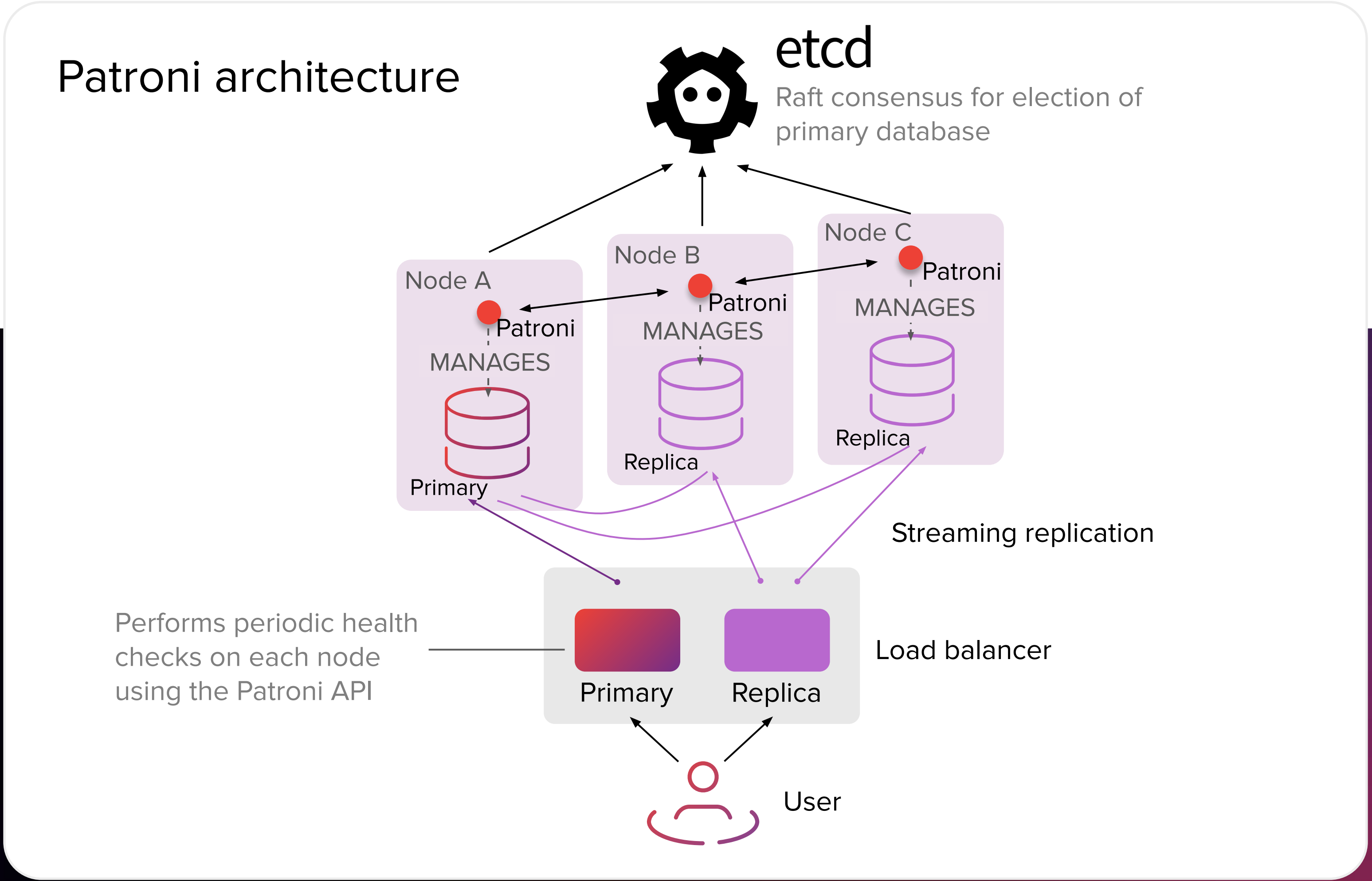
RDBMS Horizontal Scaling



RDBMS Horizontal Scaling



RDBMS Horizontal Scaling



RTB HOUSE

RDBMS Shards/Partitions

Original Table

Customer ID	First Name	Last Name	City
1	Alice	Anderson	Austin
2	Bob	Best	Boston
3	Carrie	Conway	Chicago
4	David	Doe	Denver

Vertical Shards

Customer ID	First Name	Last Name
1	Alice	Anderson
2	Bob	Best
3	Carrie	Conway
4	David	Doe

Customer ID	City
1	Austin
2	Boston
3	Chicago
4	Denver

Horizontal Shards

Customer ID	First Name	Last Name	City
1	Alice	Anderson	Austin
2	Bob	Best	Boston

Customer ID	First Name	Last Name	City
3	Carrie	Conway	Chicago
4	David	Doe	Denver

RTB HOUSE =

RDBMS Shards/Partitions

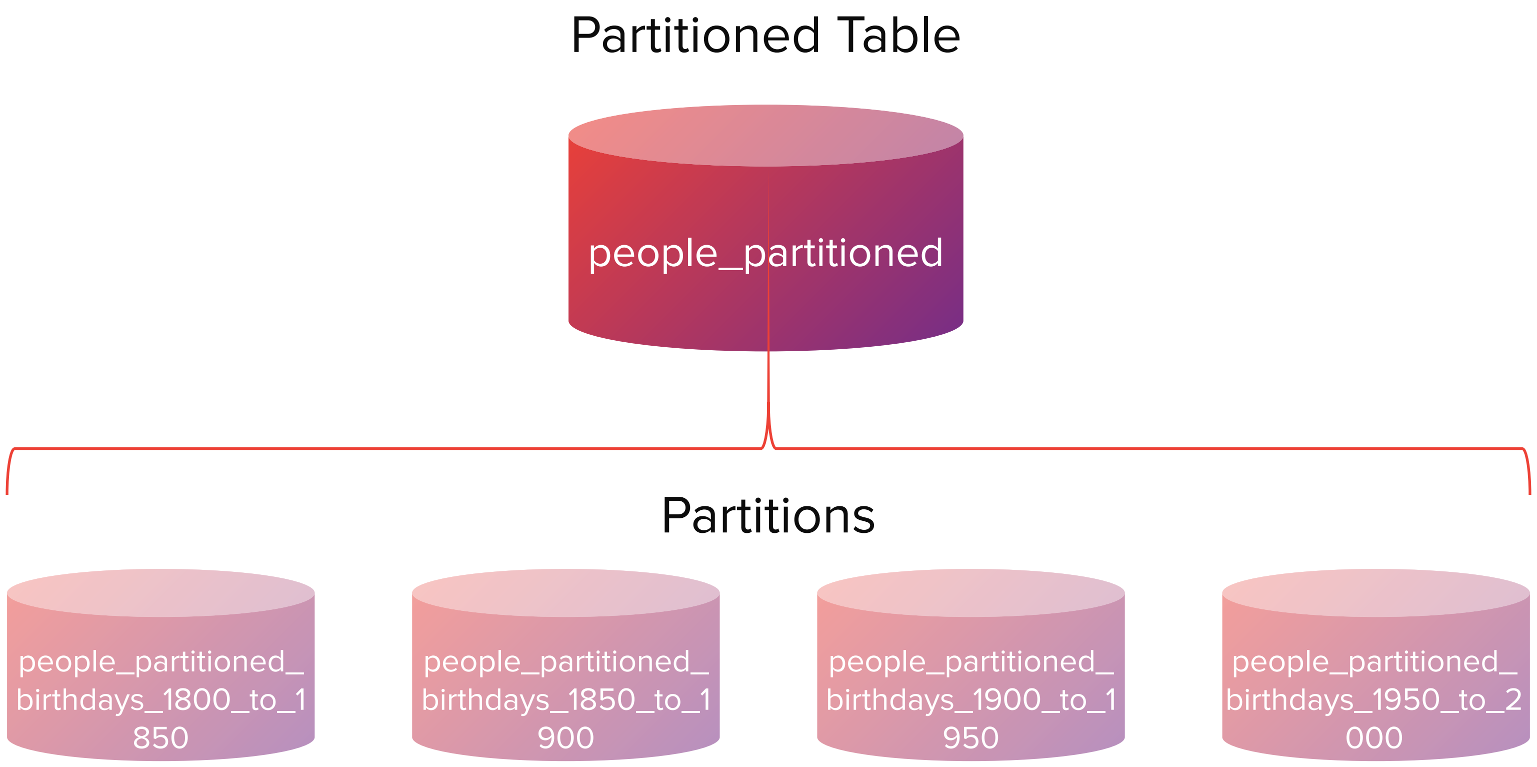
Range based –
each partition/shard allocates rows with keys in a specified range of sharding key values:

- Range read queries can be satisfied from a single partition/shard.
- Writes may be bottlenecked due to operations on one partition.
- Tendention to introduce hot spots

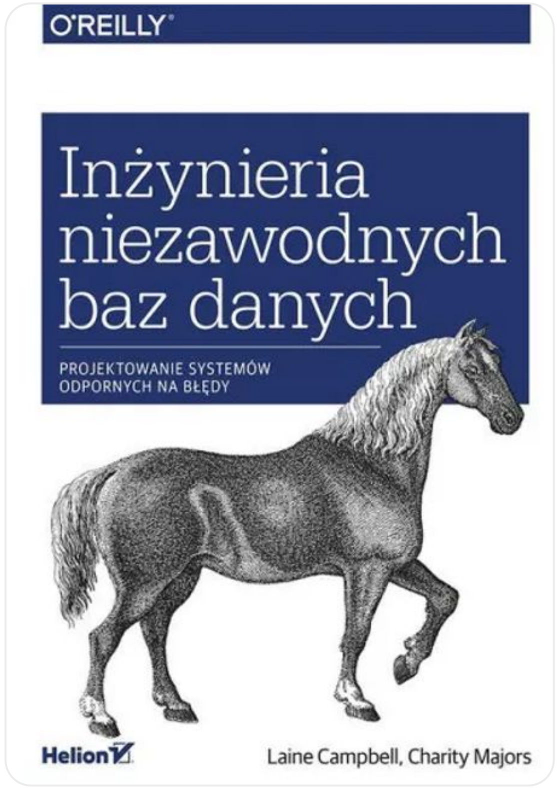
Hash based –
rows are distributed according to some hashing function applied to the sharding key:

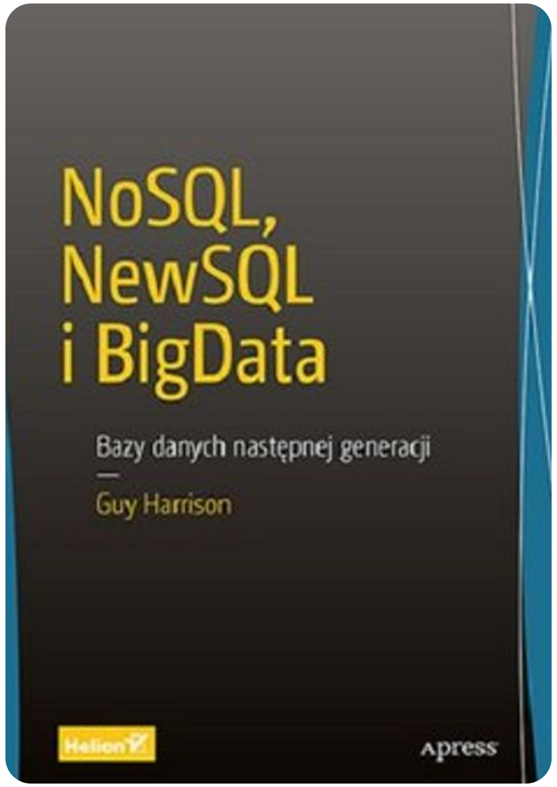
- Range read queries involve all partition/shards.
- Even distribution of records.

RDBMS Horizontal Scaling (Partitioning)


Partitioned Table
people_partitioned
Partitions
people_partitioned_birthdays_1800_to_1850
people_partitioned_birthdays_1850_to_1900
people_partitioned_birthdays_1900_to_1950
people_partitioned_birthdays_1950_to_2000

INSPIRATION


O'REILLY
Inżynieria niezawodnych baz danych
PROJEKTOWANIE SYSTEMÓW ODPORNYCH NA BŁĘDY
Helion
Laine Campbell, Charity Majors

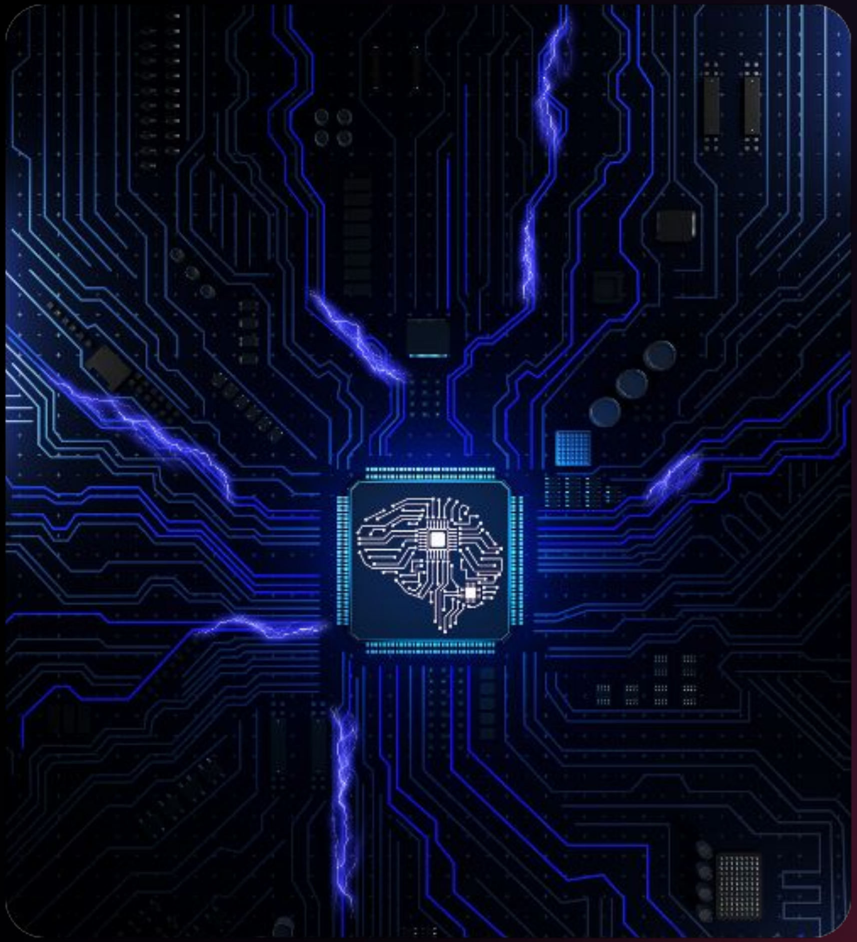

NoSQL, NewSQL i BigData
Bazy danych następnej generacji
Guy Harrison
Helion
Apress

RTB HOUSE

Summary

We have discussed:

- The aspects of replication, load balancing, high availability of RDBMS.
- Modern architecture of RDBMS.
- Common challenges.



RTB HOUSE

Thank you.

Tomasz Gintowt