Distributed Database Systems: *the case for NewSQL*

Patrick Valduriez

Principles of Distributed Database Systems

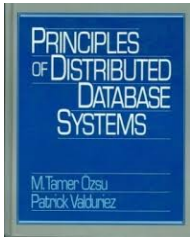
Tamer Özsu & Patrick Valduriez

The Story of the Book

Tamer Özsu

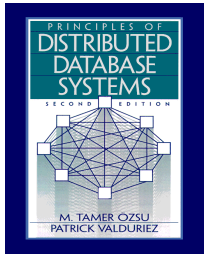
1986 Hawaii Int. Conf. on System Science: inception

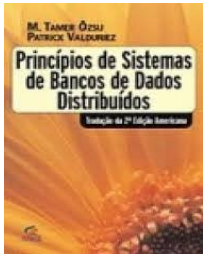
1991

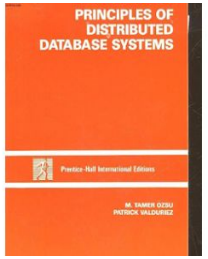


Relational databases (RDBMSs)
"In the following 10 years, centralized DBMSs would be an antique curiosity and most organizations would move towards distributed DBMSs." M. Stonebraker (1988)

1999

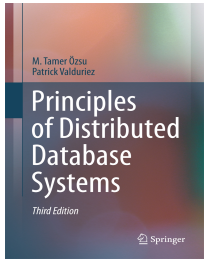


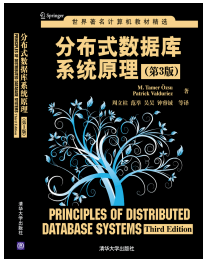




Advanced transaction models, query optimization, object data management, parallel DBMSs

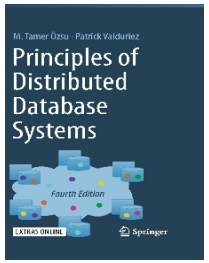
2011





Data replication, database clusters, XML, web data integration, P2P, cloud

2020



Blockchain, big data, data streaming, graph data analytics, NoSQL, NewSQL, polystores

Another Story


Martin Kersten

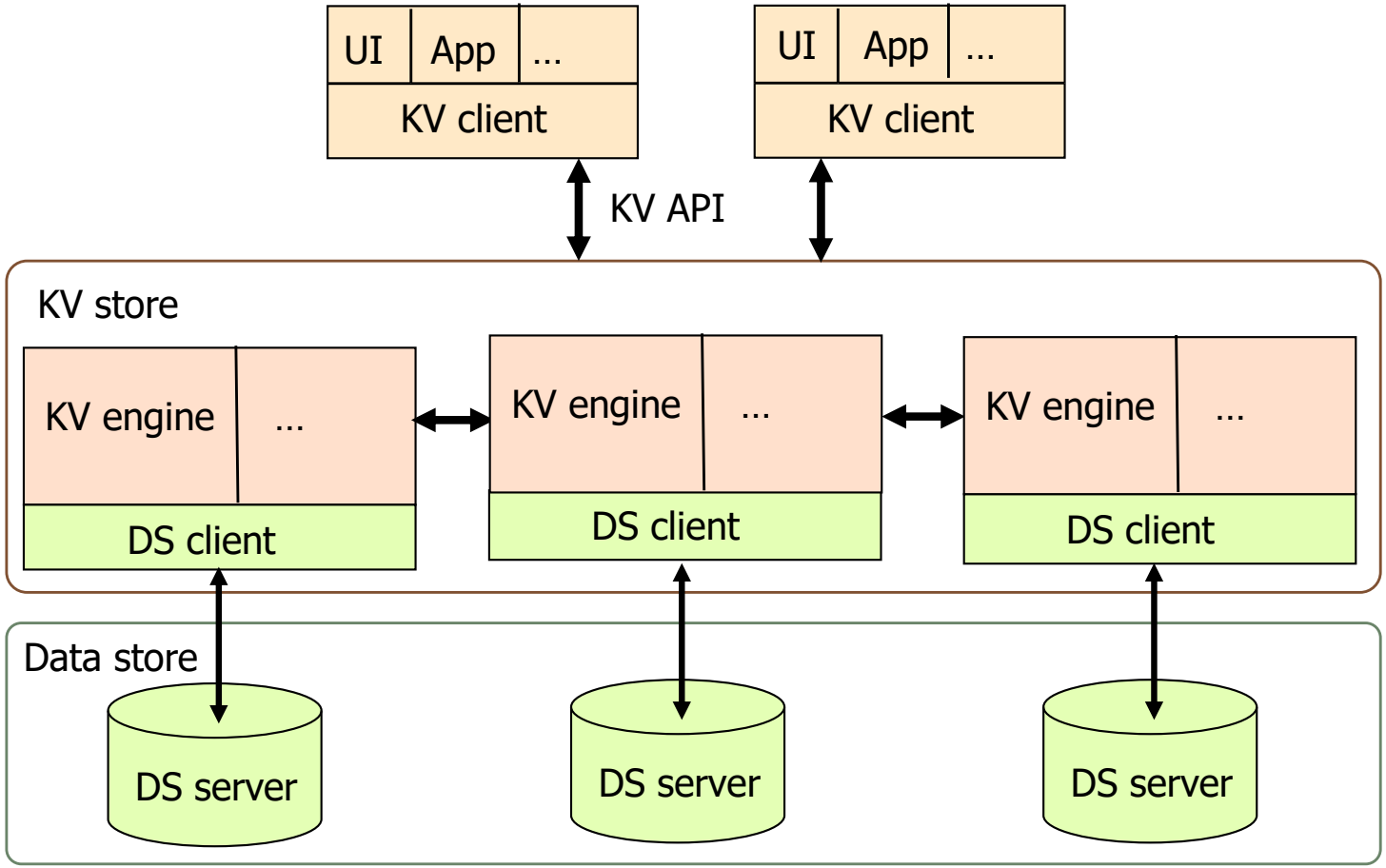
- First date: IWDM 1987, Tokyo
  - Martin Kersten et al. A Distributed, Main-Memory Database Machine (Prisma/DB project @ NL)
  - Setrag Khoshafian, Patrick Valduriez. Parallel Execution Strategies for Declustered Databases (Bubba projet @ MCC, USA)
- Then many collaborations
  - European Declarative System project, 1989-1992
  - PhD committees (both in NL and France)
  - Program committees
  - VLDB Endowment (1993-1997)
  - CoherentPaaS European Project, 2013-2016
- What impressed me most?
  - Tenacious: consistent research path from Prisma/DB to MonetDB, the pioneering main memory column store


Not
Only
SQL

NoSQL

- Different kinds
  - Key-value: DynamoDB, RockDB, Redis
  - Tabular: Bigtable, Hbase, Cassandra
  - Document (JSON): MongoDB, Coubase, CouchDB
  - Graph: Neo4J, AllegroGraph, MarkLogic
- Trade RDBMS properties
  - Full SQL, strict schema, ACID transactions
- For
  - Simplicity for designers (no schema) and programmers (simple APIs)
  - Horizontal scaling and performance
  - High availability

Key-value Store Distributed Architecture




New
SQL

NewSQL

SQL (RDBMS)

- ✓ ACID transactions
- ✓ SQL support
- ✓ Standard
- ❖ Horizontal scaling
- ❖ High availability

NoSQL

- ❖ ACID transactions
- ❖ SQL support
- ❖ Standard
- ✓ Horizontal scaling
- ✓ High availability

NewSQL

- ✓ ACID transactions
- ✓ SQL support
- ? Standard
- ✓ Horizontal scaling
- ✓ High availability

Cloud Spanner

LEANXCALE

Cockroach DB

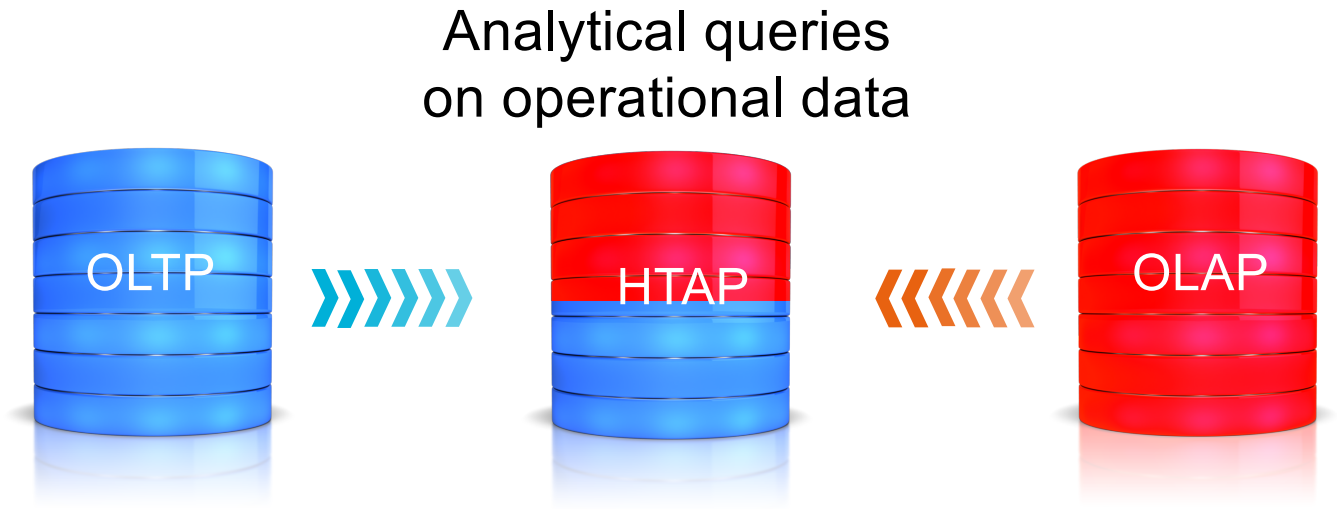
memSQL

VOLTDB

SAP HANA

splice MACHINE

HTAP*: blending OLTP & OLAP



- Advantages
  - Cutting cost of business analytics by up to 75%
  - Simpler architecture: no more ETLs/ELTs
  - Real-time analytical queries on current data

*Gartner, 2015

Use Case: Google AdWords

- Application to produce sponsored links as part of search results
  - Revenue: $134 billion in 2019
- Need to mix search queries with updates
  - To gain access to consumers, or consumer models (the probability of responding to the ad), suppliers determine the right price offer
    - Keep updating their maximum cost-per-click bid
- The AdWords database
  - 30 billion search queries per month
  - 1 billion historical search events
  - Hundreds of Terabytes

Use Case: IT Monitoring

LEANXCALE

- IT monitoring
  - The process of gathering metrics about the IT activity to ensure everything functions or will function as expected
  - Requires monitoring many systems, applications and metrics at very high frequencies
- Data management requirements
  - Real-time KPI calculation to detect root cause problems and analyze incidences
  - Combine historical and current data for drill-down, forecasting and reporting
- Solution
  - Fast data ingestion (as with NoSQL) and analytical processing (as with SQL)

Main Techniques for NewSQL

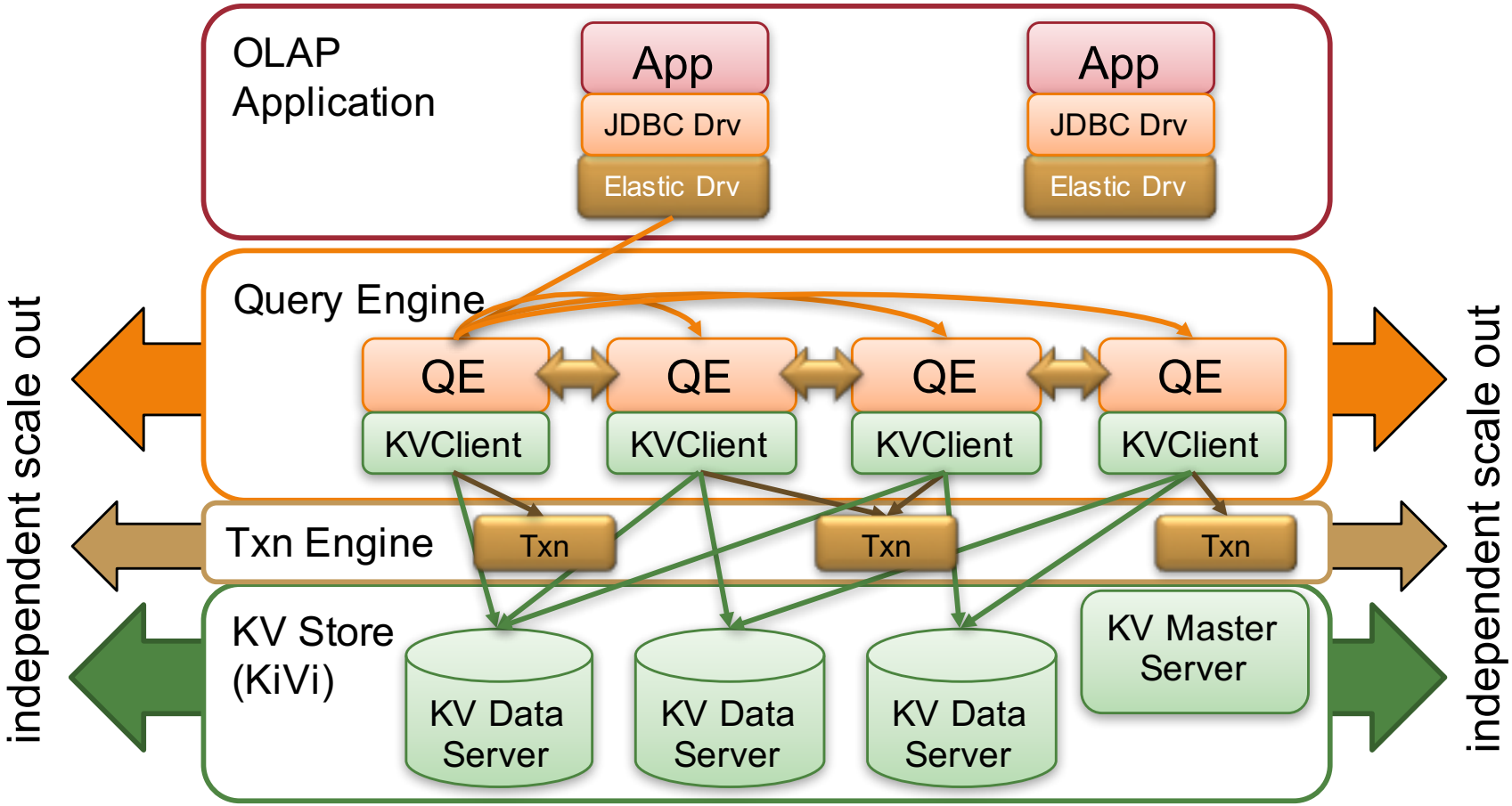
- From SQL
  - Data analytics
  - Parallel, in-memory query processing
  - Column-store
- From NoSQL
  - Key-value storage and access
  - Horizontal and vertical data partitioning
  - Fault-tolerance, failover and synchronous replication
- New
  - Scalable transaction management
  - Polyglot language and polystore
    - Access to SQL, NoSQL and HDFS data stores

LEANXCALE

Ricardo Jimenez-Peris

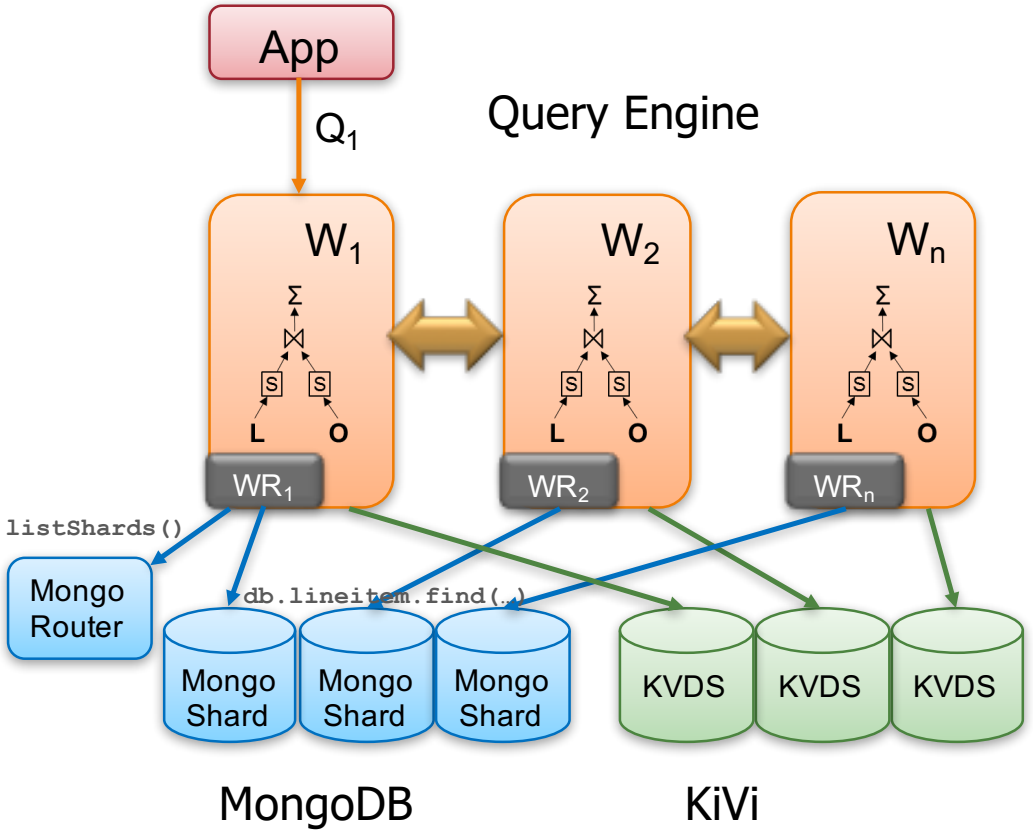
- SQL DBMS
  - Polyglot language with JSON support
- Relational key-value store (KiVi)
  - Fast, parallel data ingestion
  - Online aggregation with aggregation tables
- OLAP Query Engine
  - Intra-query intra-operator parallelism
  - Polystore access: HDFS, MongoDB, Hbase, …
- Ultra-scalable transaction processing
  - SQL snapshot isolation level

Architecture



Parallel Polystore Query Processing

- LeanXcale Query Engine
  - CloudMdsQL polystore[1]
- Exploit data sharding in data stores[2]
  - Intra-operator parallelism
  - Optimization
    - Select pushdown, bindjoin, etc.



1. B. Kolev, C. Bondiombouy, P. Valduriez, R. Jiménez-Peris, R. Pau, J. Pereira. The CloudMdsQL Multistore System. SIGMOD 2016.
2. B. Kolev, O. Levchenko, E. Pacitti, P. Valduriez, R. Vilaça, R. Gonçalves, R. Jiménez-Peris, P. Kranas. Parallel Polyglot Query Processing on Heterogeneous Cloud Data Stores with LeanXcale. IEEE Big Data, 2018.

Scalable Transaction Processing*

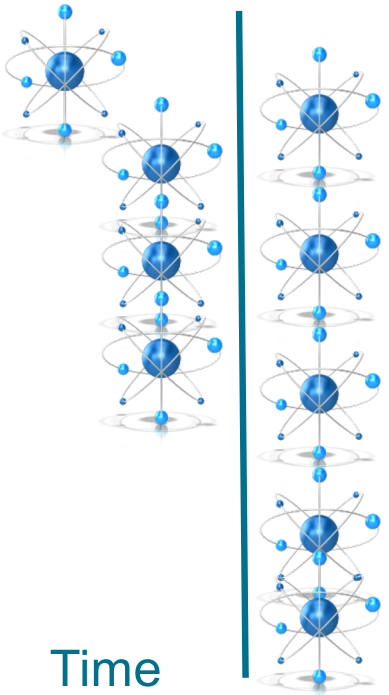
LEANXCALE

Traditional approach

Single-node bottleneck

Time

vs



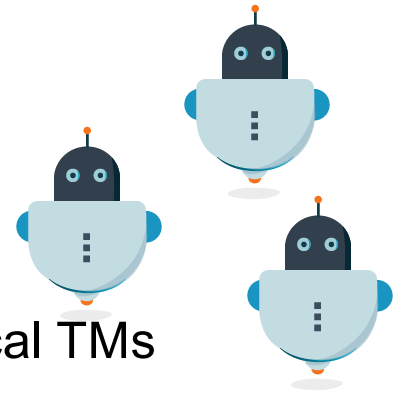
Time

Processes & commits transactions **in parallel**

Provides a **consistent view**

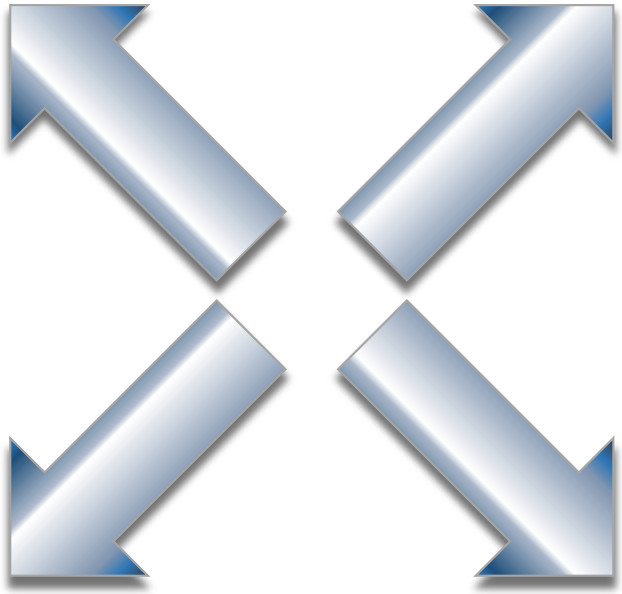
* R. Jimenez-Peris, M. Patiño-Martinez. System and method for highly scalable decentralized and low contention transactional processing. Priority date: 11th Nov. 2011. European Patent #EP2780832, US Patent #US9,760,597.

Scaling ACID Properties



Local TMs

Atomicity



Conflict managers

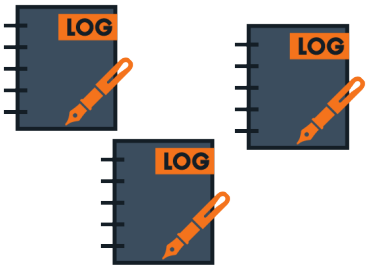
Isolation Writes

Isolation Reads

Snapshot server

Commit sequencer

Durability

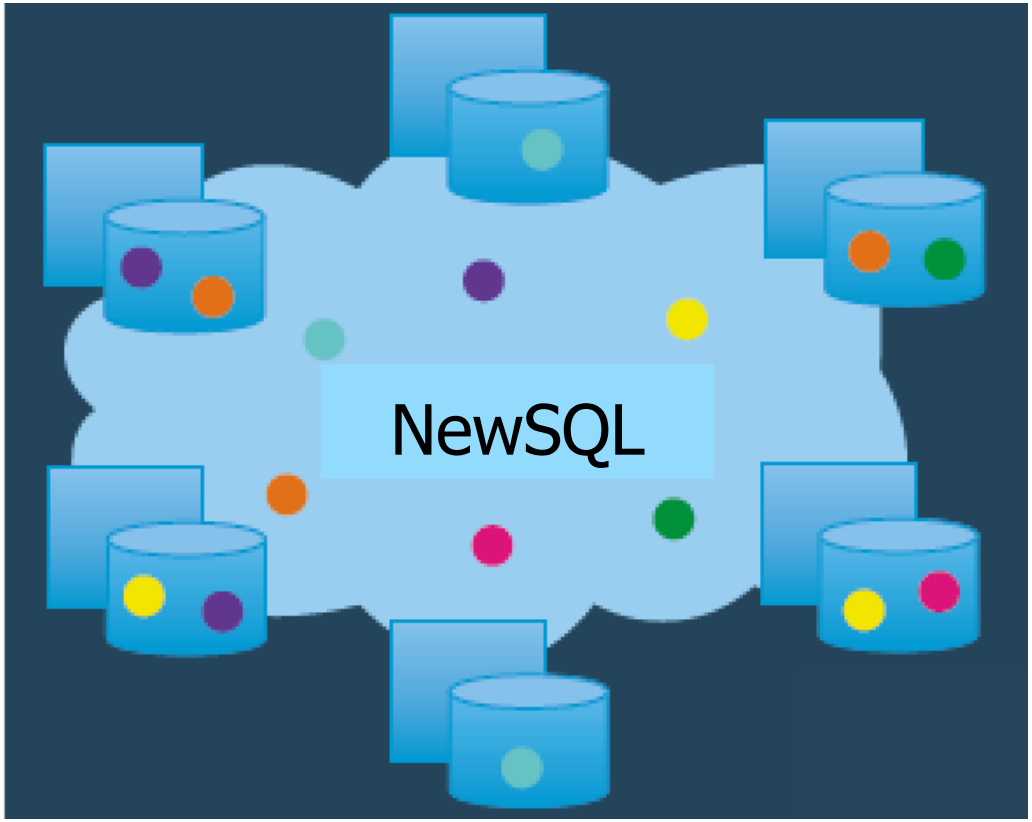


Loggers

Transaction Mgt Principles

- Separation of commit from the visibility of committed data
- Proactive pre-assignment of commit timestamps to committing transactions
- Detection and resolution of conflicts before commit
- Transactions can commit in parallel because:
  - They do not conflict
  - They have their commit timestamp already assigned that will determine their serialization order
  - Visibility is regulated separately to guarantee the reading of fully consistent states

Conclusion



Main Takeaways

- Lessons learned
  - Distributed database systems passed the test of time (centralized DBMSs are an antique curiosity)
  - Clean principles, the basis for many variations: SQL, NoSQL, NewSQL
- NewSQL, the future of SQL?
  - HTAP becoming a major workload
  - Many research opportunities
    - Integration with polystores, streaming, machine learning, …
    - Benchmarking