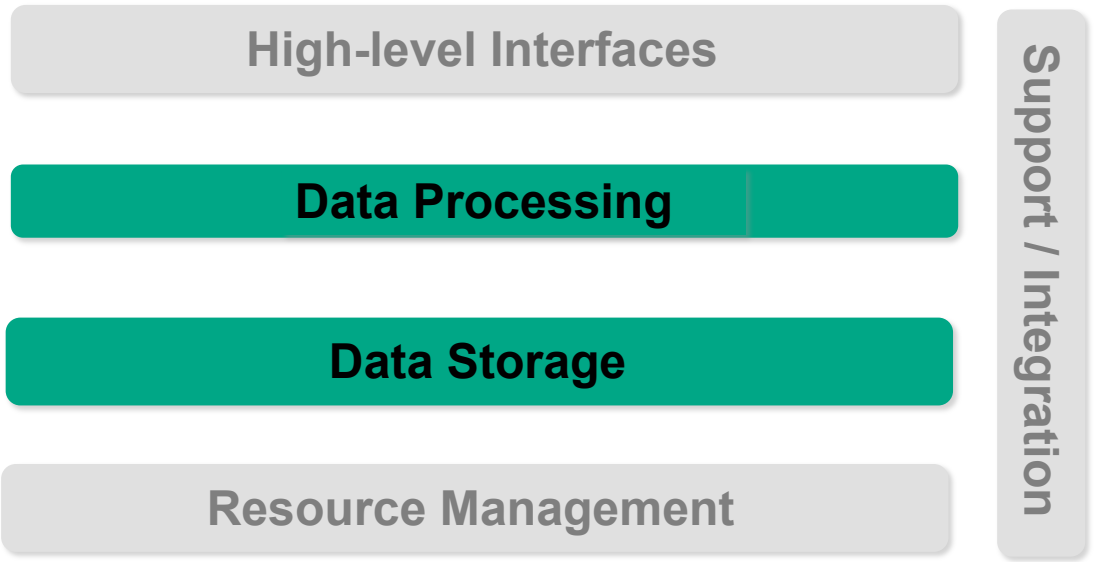
Università degli Studi di Roma “Tor Vergata”
Dipartimento di Ingegneria Civile e Ingegneria Informatica

NewSQL Databases

Corso di Sistemi e Architetture per Big Data
A.A. 2017/18

Valeria Cardellini

The reference Big Data stack



Relational database services

- RDBMS pros:
  - ACID transactions
  - Relational schemas (and schema changes without downtime)
  - SQL queries
  - Strong consistency
- RDBMS cons:
  - Lack of horizontal scalability, to hundreds or thousands of servers

NewSQL databases

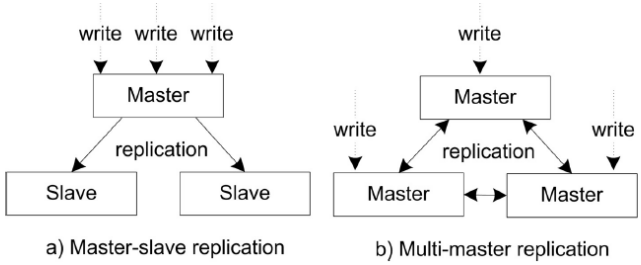
- How to build a relational database service that is both strongly consistent and horizontally scalable?
- **NewSQL**: a class of modern RDBMSs
- Goals
  - Provide the same scalable performance of NoSQL systems for OLTP read-write workloads while maintaining ACID guarantees of traditional RDMSs
  - Support SQL

NewSQL examples

- Google's Spanner
  - Also available as cloud service in Google Cloud Platform: Cloud Spanner https://cloud.google.com/spanner/
- Google's F1
  - Built on top of Spanner
- VoltDB
  - And H-Store, its research prototype predecessor http://hstore.cs.brown.edu
  - Developed by M. Stonebraker (2015 ACM Turing award)
- Clustrix
  - Closed source
- NuoDB
  - Closed source, no support for stored procedures

Replication in NewSQL

- Multi-master or master-less schemes
  - Any node can receive update statements



- VoltDB and Clustrix
  - A transaction/session manager receives the updates, which are forwarded to all replicas and executed in parallel
- Google Spanner
  - Uses Paxos state machine replication to guarantee that a sequence of commands will be executed in the same order in all the replica nodes

Spanner

- Motivations:
  - “We provide a database instead of a key-value store to make it easier for programmers to write their applications”
  - “We consistently received complaints from users that Bigtable can be difficult to use for some kinds of applications”

What is Spanner

- Wide-area distributed multi-version database
  - General-purpose transactions (ACID)
  - SQL query language (slightly modified)
  - Schematized semi-relational tables
    - A hierarchical approach to grouping tables that allows Spanner to co-locate related data into directories that can be easily stored, replicated, locked, and managed
- Running in production
  - Storage for Google’s ads data
  - Replaced a sharded MySQL database

Spanner overview

- Feature: lock-free distributed read transactions
- Property: external consistency of distributed transactions
  - First system at global scale
- Implementation: integration of concurrency control, replication, and 2PC
  - Correctness and performance
- Enabling technology: a new API Time called TrueTime
  - Interval-based global time
  - Based on hardware-assisted time synchronization using GPS clocks and atomic clocks
  - Very low overhead (14ms introduced by Spanner for data centers at 1 ms network distance)

Concurrency control in Spanner

- Hybrid approach
  - Read-write transactions are implemented through read-write locks, but read-only transactions are lock-free
- Why is it possible?
  - Spanner stores multiple versions of data, and a read transaction is basically a read at a “safe” timestamp

VoltDB

- In-memory database with shared nothing architecture
- Starting point
  - Open source RDBMS ran on memory-based file system
    - Over 80% of time spent on page buffer management, index management, and concurrency management
    - Only 12% of time spent doing the real work
  - Lead to H-Store
- Features
  - Horizontal scale-out on commodity hardware with linear scalability
  - Fully ACID compliant
  - High concurrency
  - Reliable disk persistence (both asynchronous and synchronous command logging)
  - High availability

VoltDB

- Tables are partitioned over multiple servers, and clients can call any server
  - Transparent distribution but the user can choose the sharding attribute
- Selected tables can be replicated over servers, e.g. for fast access to read-mostly data
- Shards are replicated, so that data can be recovered in the event of a node crash
- Database snapshots are also supported, continuous or scheduled

VoltDB and concurrency control

- Alternative design based on two assumptions
  - Assumption 1: total available memory is large enough to store the entire data store
  - Assumption 2: all user transactions are short-lived and can be very efficiently executed over in-memory data
- Then, all transactions are executed sequentially in a single-threaded, lock-free environment

References

- Golinger et al., “Data management in cloud environments: NoSQL and NewSQL data stores”, *J. Cloud Comp.*, 2013. http://bit.ly/2oRKA5R
- Corbett et al., “Spanner: Google’s globally distributed database”, *OSDI 2012*. http://bit.ly/2nyJBrb
- Stonebraker and Weisberg, “The VoltDB main memory DBMS”, 2013. http://bit.ly/2okw837