VoltDB The NewSQL database for high velocity applications

New Age Transactional Systems – Not Your Grandpa's OLTP

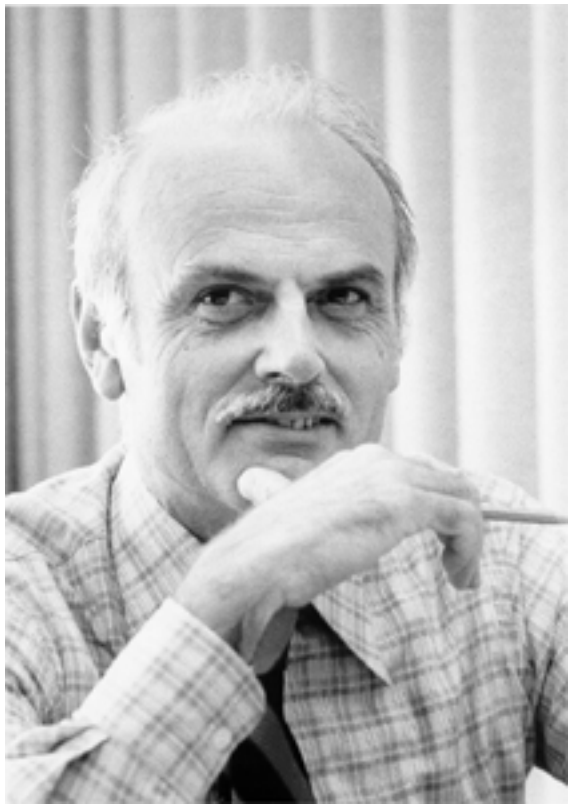
Ryan Betts
Sr. Software Engineer, VoltDB
rbetts@voltdb.com / @ryanbetts

QCon

Who's making fun of Grandpa's DB?

QCon

We got your back, Mr. Codd.



Are there crystals involved?

QCon

New age OLTP

NoSQL

NewSQL

More than marketing?

NewSQL

QCon

"…shorthand for various new scalable high performance db vendors"

Matthew Aslett / The 451 group
April, 2011

http://blogs.the451group.com/information_management/2011/04/06/what-we-talk-about-when-we-talk-about-newsql/

NewSQL

QCon

"…relational semantics with NoSQL scalability"®

Me

November 16th.

Relational

ACID
Transactions
SQL
Schema

NoSQL

Cloud deployable
Distributed data
Scale-out
Low cost profile

NewSQL v. Legacy RDBMS

Tradeoffs differ by vendor

Analysts. Me. Whoever.

QCon

Maybe famous academics have more credibility?

SIGMOD 2008

QCon

OLTP Through the Looking Glass, and What We Found There

Stavros Harizopoulos
Daniel J. Abadi
Samuel Madden
Michael Stonebraker

http://nms.csail.mit.edu/~stavros/pubs/OLTP_sigmod08.pdf

Brilliant!

QCon

Computers are different from 1970's

- Cheaper
- Lots of main memory
- Multi-core

Applications are different from 1970's

- Web and middleware: not data entry
- Latency sensitive "real time" apps.

Overhead in the Shore DB

QCon

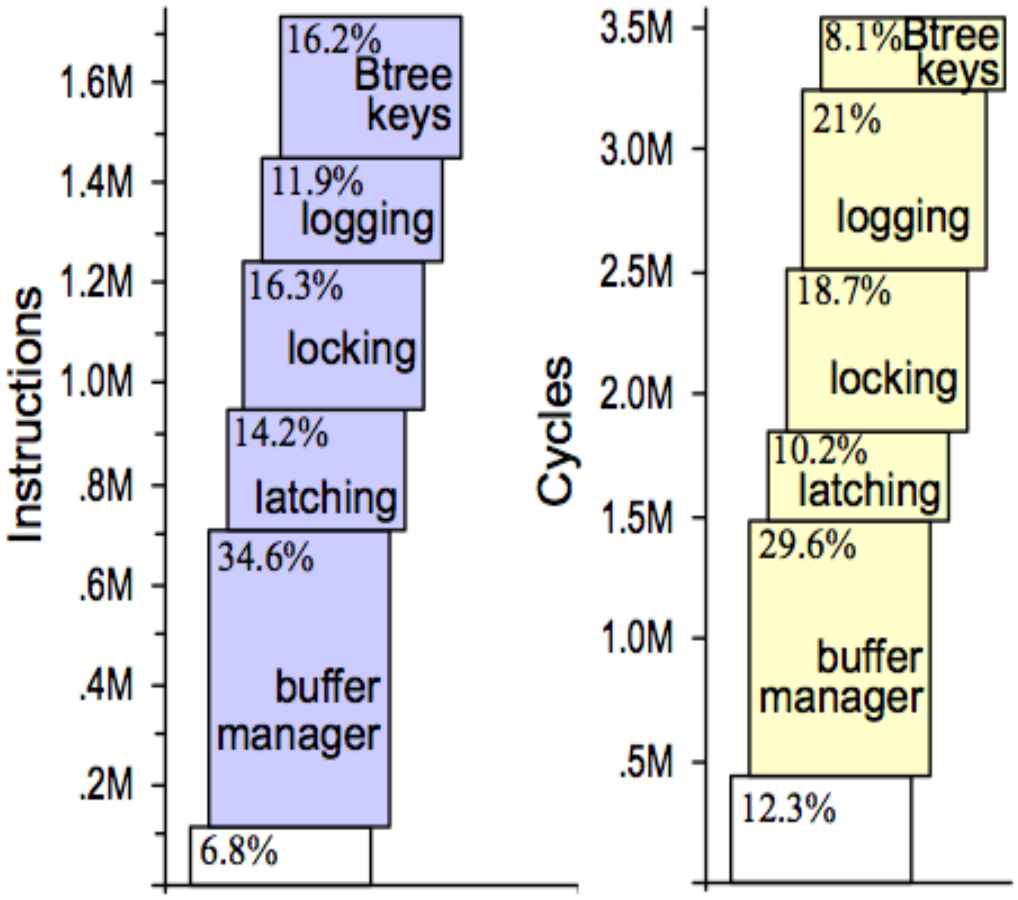


Figure 8. Instructions (left) vs. Cycles (right) for New Order.

Legacy overhead

Btrees
Logging
Locking
Latching
Buffer Cache

TPC-C NewOrder and Payment

QCon

Full overhead (in memory, no log fsync):
1,700 TPS.

Optimal (measured via prototype):
46,500 TPS.

VoltDB: 1M SQL statements / sec.
50k-180k TPS per node on current hardware

Are these kids on to something?

QCon

Many alternative architectures

Memory resident databases
Clustered databases
Log-less databases
Single threaded databases
Transaction-less databases
Eventually consistent systems

Analysts. Me. Academics.

QCon

What's the rest of world have to say about NewSQL and New age OLTP?

Why?

Why.

QCon

Contemporary workloads exceed the capability of legacy database architectures

Do you have this problem?

QCon

You have a few choices.

Do you have this problem?

QCon

Choice #1

Ignore it. Maybe it just isn't that important.

Do you have this problem?

QCon

Choice #2

Shard a legacy system. You might regret this choice.

Manage your own sharding.
Your own transactions.
Operational overhead many nodes.
Poor per-node performance.

Do you have this problem?

QCon

Choice #3

NoSQL: Free coffee cups, laptop stickers
AND it shards for me!

Do you have this problem?

QCon

Choice #4

NewSQL: I like new things. Sign me up.

Do you have this problem?

QCon

NewSQL vs. NoSQL: what are the important decision factors.

NoSQL strengths

QCon

Raw availability vs. consistency and (ACID) durability.

Journal oriented or commutative writes.

Schema flexibility (continuously deployed web apps?)

NewSQL strengths

QCon

ACID semantics.

Especially multi-key ACID.

Transactions are necessary.

Rich queries. Non-trivial atomic operations.

Note the commonalities

QCon

Cloud friendly.

Scale-out. Horizontal scaling.

Low cost profile.

New age OLTP use cases

QCon

Example workloads that favor the NewSQL side of this decision matrix

Use case: Online Gaming

QCon

Game state tracking is often transactional.

State must be durable: what was observed as done must stay done.

"Sharding games detracts from the universality and player experience."

Use case: Advertising Tech

QCon

Near real time (not batched) A/B testing:

- Stream of campaign interactions (clicks).
- Aggregated multi-key reads of results.
- Real time feedback in to ad composition.

Ad display decision making:

- Input demographics & run business logic.
- Record choices.

Use case: Financial Trade Apps

QCon

Process 100k to 1M+ inputs per second:

- Calculate outstanding positions/risk.
- Measure portfolio compliance (alert).
- Measure performance by sub-groups (report).

Real time web apps

QCon

Durable session caches.

Reduce latency of feedback to web-tier.

Web -> log -> OLTP analytics -> Web

Telco

QCon

Near real time account status.

- Prepaid account alerting.
- Roaming / data plan alerting.
- Real time account balance inquiries.

Network monitoring

QCon

Endpoint monitoring for attack profiles.

Monitoring traffic flows.

Operational dashboarding.

Sensor readings

QCon

De-dupe multiple readings.

Filter.

Alert on location.

Cross sensor aggregates.

Leaderboarding.

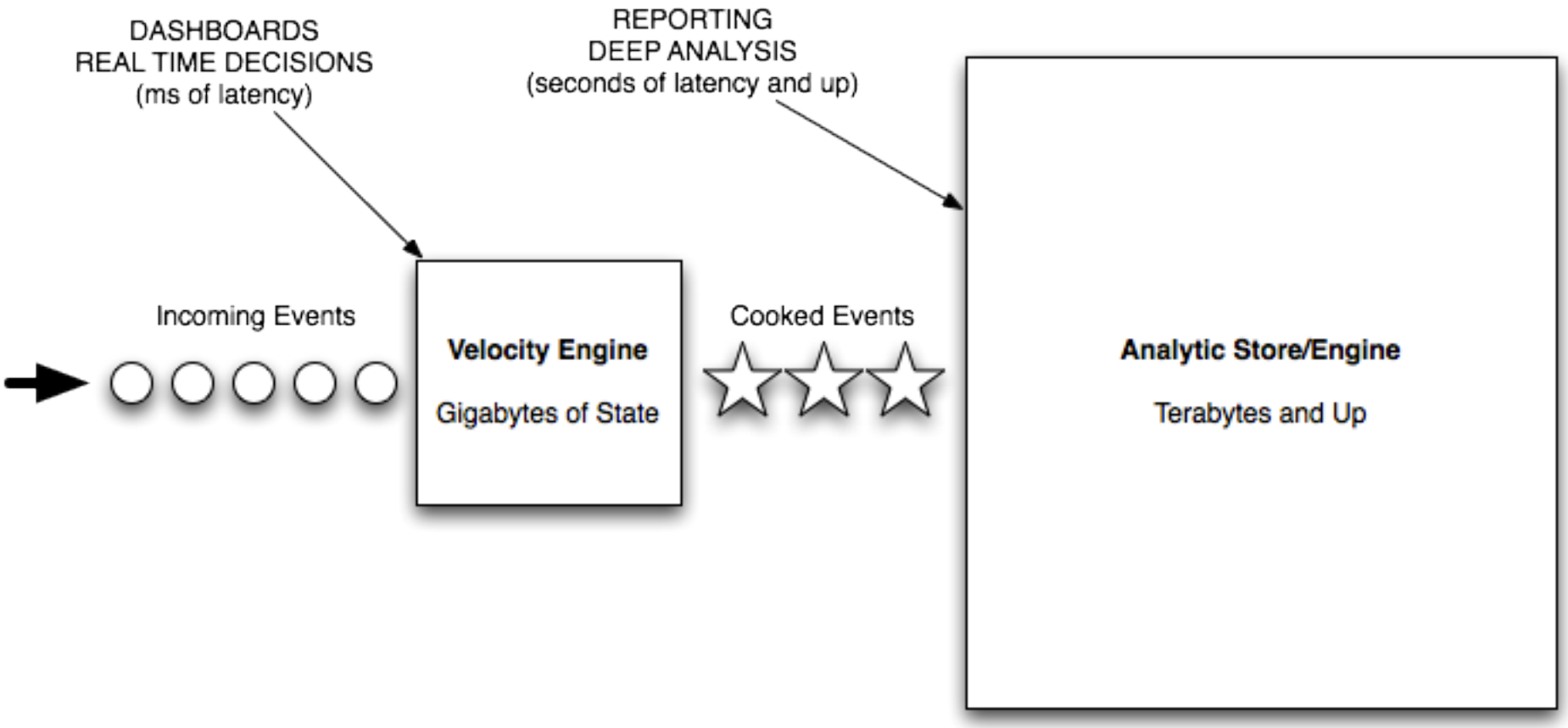
Application Use Cases

QCon

	Data Source	High-frequency operations	Lower-frequency operations
Financial trade monitoring	Capital markets	Write/index all trades, store tick data	Show consolidated risk across traders
Telco call data record management	Call initiation request	Real-time authorization	Fraud detection/analysis
Website analytics, fraud detection	Inbound HTTP requests	Visitor logging, analysis, alerting	Traffic pattern analytics
Online gaming micro transactions	Online game play	Rank scores: •Defined intervals •Player "bests"	Leaderboard lookups
Digital ad exchange services	Real-time ad trading systems	Match form factor, placement criteria, bid/ask	Report ad performance from exhaust stream
Wireless location-based services	Mobile device location sensor	Location updates, QoS, transactions	Analytics on transactions

A picture is worth a thousand bullets…

QCon



The Giant Scoreboard in the Sky

Counting is Hard: American Idol Edition

QCon

N contestants on some reality talent show

Vote over the phone and count votes per contestant

- Option 1: Accept that you're not going to be perfectly accurate, and that it might be difficult to bound how not accurate.
- Option 2: Buffer batches of votes for a while to amortize the cost of using external locks. The real-time count for a contestant is a count(*) on live buffers plus the roll up of prev. buffers.
- Option 3: Support isolated, atomic read-then-update for fault-tolerant counters (at very high throughput).

Counting is Hard: Requirements change

QCon

N contestants on some reality talent show

Vote over the phone and count votes per contestant

Limit voters to X number of votes

BEGIN (PHONE, CONTESTANT)

Check how many votes a phone number has.
If too many, return.
Otherwise atomically increment the votes for the phone number and for the contestant.

COMMIT

… a lot

QCon

- N contestants on some reality talent show.
- Vote over the phone and count votes per contestant.
- Limit voters to X number of votes.
- Allow voters who've signed up for the email list to vote twice as many times.
- Don't allow voters to vote twice in the same minute.
- Send voters an SMS text message after their 3rd vote. DON'T SEND TWO.
- Provide a live dashboard of contestant scores by geo.
- Record votes past the limits, but don't count them.

User feedback

QCon

What do you hear when the right problem is paired to the right tool?

User feedback

QCon

If you need transactions, building them yourself is painful.

Using a system with transactions will cut development time and increase correctness.

User feedback

QCon

Per-Node performance matters.

Can support naïve solutions.

MySQL + memcached + tuning + SSD + cleverness gets you a lot. But a NewSQL system might be nearly idle on the same workload.

User feedback

QCon

Per-Node performance matters.

Enables cloud deployment. More predictable I/O. Reduces cloud leasing costs.

Enables you to balance scaling up vs. out.

NewSQL product similarities?

QCon

Techniques differ widely. From single threaded + transactional procedures to MVCC to alternate MySQL storage engines.

Common: (1) SQL as query language. (2) ACID transactions. (3) Non-blocking concurrency control. (4) per-node performance (5) horizontal scale-out.

Conclusions

QCon

NewSQL can be differentiated from NoSQL and from legacy RDBMS.

Some workloads favor NewSQL:

- Rich query-ability
- ACID transactions
- High velocity inputs
- Multi-key aggregation

Questions & Answers

Thanks!

rbetts@voltdb.com / @ryanbetts