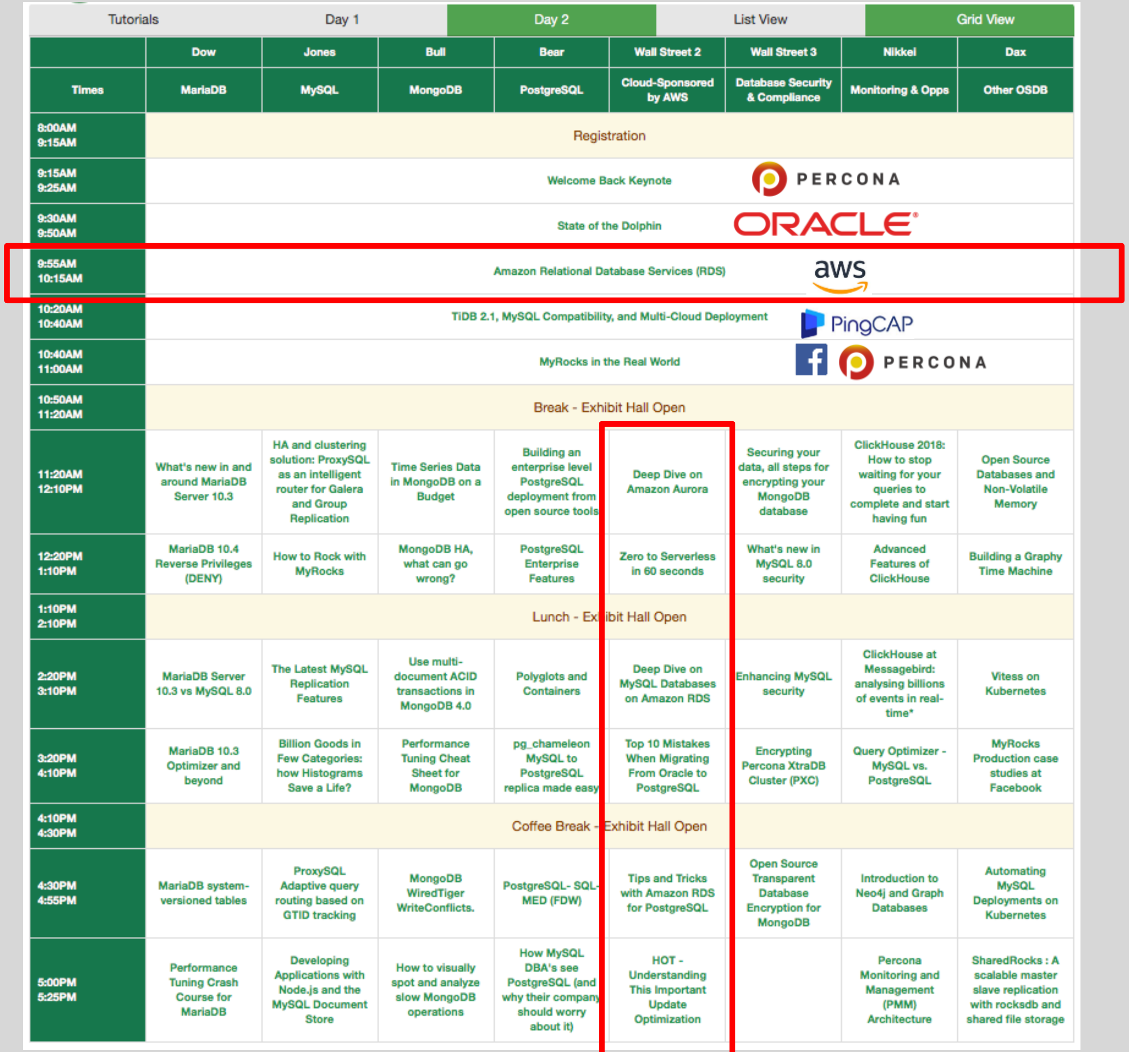
Deep Dive on Amazon Aurora

Kevin Jernigan
kmj@Amazon.com
Principal Product Manager, Amazon Aurora
Amazon Web Services

OPEN SOURCE DATABASE CONFERENCE

PERCONA
LIVE EUROPE
FRANKFURT



	Dow	Jones	Bull	Bear	Wall Street 2	Wall Street 3	Nikkel	Dax
Times	MariaDB	MySQL	MongoDB	PostgreSQL	Cloud-Sponsored by AWS	Database Security & Compliance	Monitoring & Opps	Other OSDB
8:00AM 9:15AM	Registration							
9:15AM 9:25AM	Welcome Back Keynote PERCONA							
9:30AM 9:50AM	State of the Dolphin ORACLE							
9:55AM 10:15AM	Amazon Relational Database Services (RDS) aws							
10:20AM 10:40AM	TiDB 2.1, MySQL Compatibility, and Multi-Cloud Deployment PingCAP							
10:40AM 11:00AM	MyRocks in the Real World PERCONA							
10:50AM 11:20AM	Break - Exhibit Hall Open							
11:20AM 12:10PM	What's new in and around MariaDB Server 10.3	HA and clustering solution: ProxySQL as an intelligent router for Galera and Group Replication	Time Series Data in MongoDB on a Budget	Building an enterprise level PostgreSQL deployment from open source tools	Deep Dive on Amazon Aurora	Securing your data, all steps for encrypting your MongoDB database	ClickHouse 2018: How to stop waiting for your queries to complete and start having fun	Open Source Databases and Non-Volatile Memory
12:20PM 1:10PM	MariaDB 10.4 Reverse Privileges (DENY)	How to Rock with MyRocks	MongoDB HA, what can go wrong?	PostgreSQL Enterprise Features	Zero to Serverless in 60 seconds	What's new in MySQL 8.0 security	Advanced Features of ClickHouse	Building a Graphy Time Machine
1:10PM 2:10PM	Lunch - Exhibit Hall Open							
2:20PM 3:10PM	MariaDB Server 10.3 vs MySQL 8.0	The Latest MySQL Replication Features	Use multi-document ACID transactions in MongoDB 4.0	Polyglots and Containers	Deep Dive on MySQL Databases on Amazon RDS	Enhancing MySQL security	ClickHouse at Messagebird: analysing billions of events in real-time*	Vitess on Kubernetes
3:20PM 4:10PM	MariaDB 10.3 Optimizer and beyond	Billion Goods in Few Categories: how Histograms Save a Life?	Performance Tuning Cheat Sheet for MongoDB	pg_chameleon MySQL to PostgreSQL replica made easy	Top 10 Mistakes When Migrating From Oracle to PostgreSQL	Encrypting Percona XtraDB Cluster (PXC)	Query Optimizer - MySQL vs. PostgreSQL	MyRocks Production case studies at Facebook
4:10PM 4:30PM	Coffee Break - Exhibit Hall Open							
4:30PM 4:55PM	MariaDB system-versioned tables	ProxySQL Adaptive query routing based on GTID tracking	MongoDB WiredTiger WriteConflicts.	PostgreSQL- SQL-MED (FDW)	Tips and Tricks with Amazon RDS for PostgreSQL	Open Source Transparent Database Encryption for MongoDB	Introduction to Neo4j and Graph Databases	Automating MySQL Deployments on Kubernetes
5:00PM 5:25PM	Performance Tuning Crash Course for MariaDB	Developing Applications with Node.js and the MySQL Document Store	How to visually spot and analyze slow MongoDB operations	How MySQL DBA's see PostgreSQL (and why their company should worry about it)	HOT - Understanding This Important Update Optimization		Percona Monitoring and Management (PMM) Architecture	SharedRocks : A scalable master slave replication with rocksdb and shared file storage

Agenda

- Aurora overview
- Performance improvements
- Availability improvements
- Recent innovations

“Intuit invests significantly to own and operate high-end commercial databases underpinning our business. Until now, there just wasn’t a real alternative to obtain the **reliability** and **performance** our customers need. Amazon Aurora is a game-changer for us: providing the performance and availability features that rival expensive on-premises databases and SANs at a **significantly lower price point**. The RDS **management capabilities** on top of Amazon Aurora will allow us to focus our resources and energy on what matters most – building great applications and delighting our customers.”

Troy Otillio, Director, Public Cloud, Intuit

What is Amazon Aurora?

Database reimagined for the cloud

- ☑ **Speed** and **availability** of high-end commercial databases
- ☑ **Simplicity** and **cost-effectiveness** of open source databases
- ☑ Drop-in **compatibility** with MySQL and PostgreSQL
- ☑ Simple **pay as you go** pricing

Delivered as a **managed** service

Re-imagining the relational database

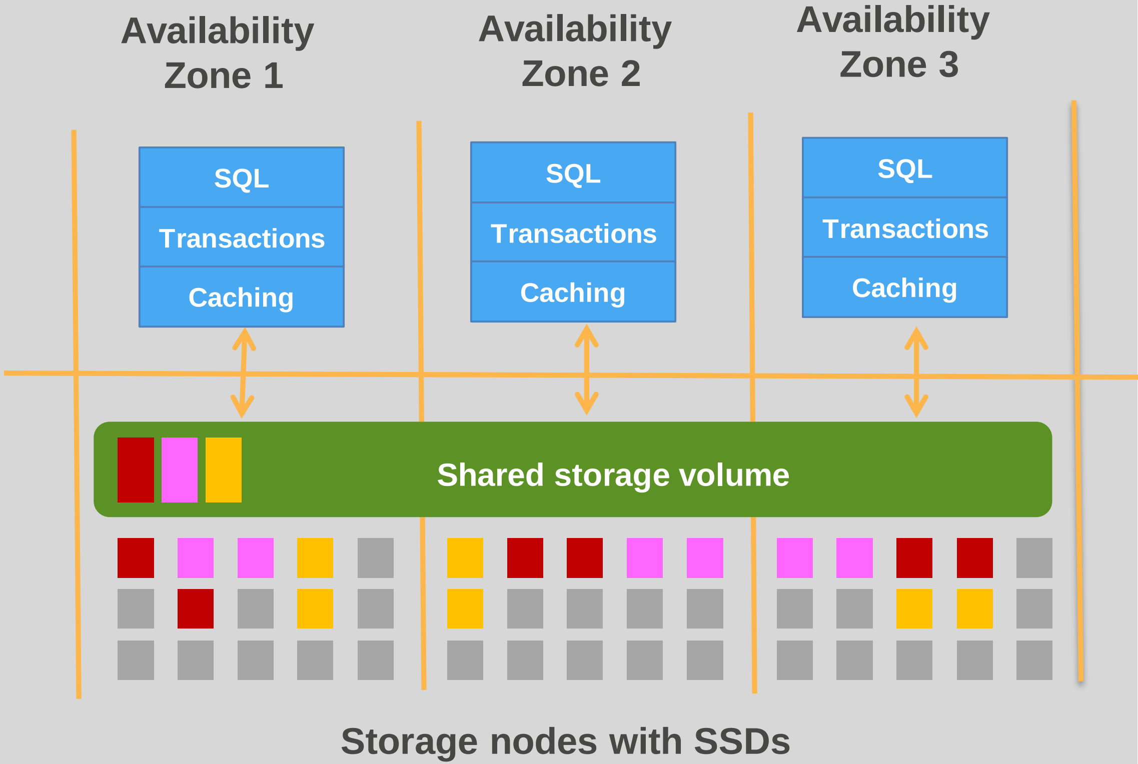
1. Scale-out, distributed architecture

2. Service-oriented architecture leveraging AWS services

3. Automate administrative tasks – fully managed service

Scale-out, distributed architecture

- Purpose-built log-structured distributed storage system designed for databases
- Storage volume is striped across hundreds of storage nodes distributed over 3 different availability zones
- Six copies of data, two copies in each availability zone to protect against AZ+1 failures
- Plan to apply same principles to other layers of the stack



Leveraging AWS services

Lambda

Invoke Lambda functions from stored procedures/triggers

S3

Load data from/ Select into S3, store snapshots and backups in S3

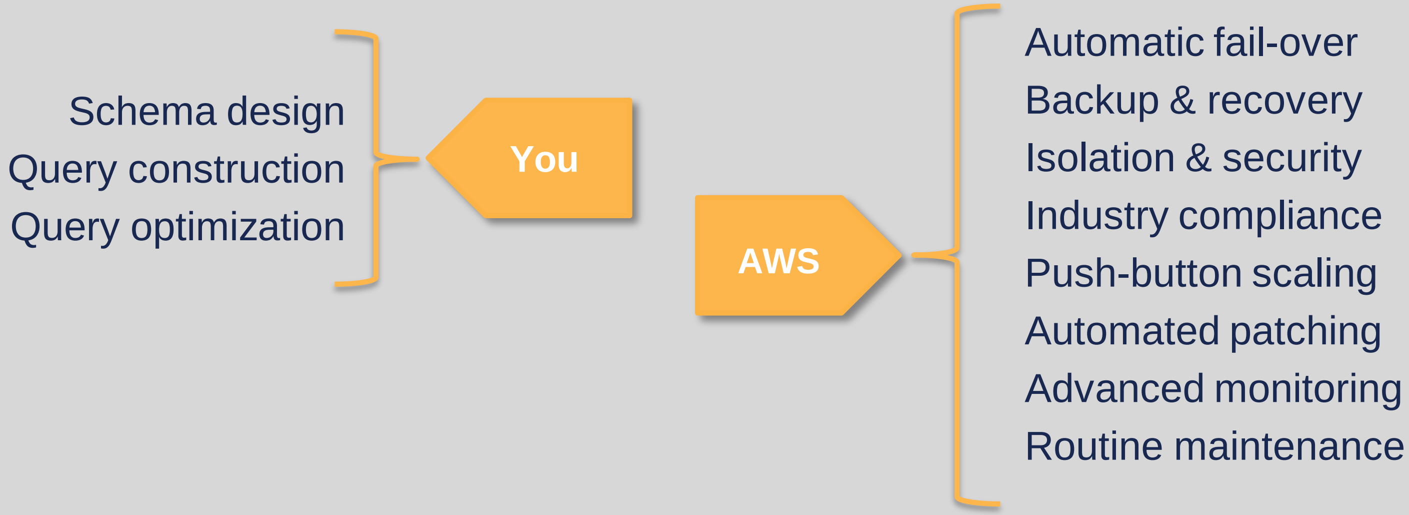
IAM

Use IAM roles to manage database access control

CloudWatch

Upload systems metrics and database logs

Automate administrative tasks



Schema design

Query construction

Query optimization

You

AWS

Automatic fail-over

Backup & recovery

Isolation & security

Industry compliance

Push-button scaling

Automated patching

Advanced monitoring

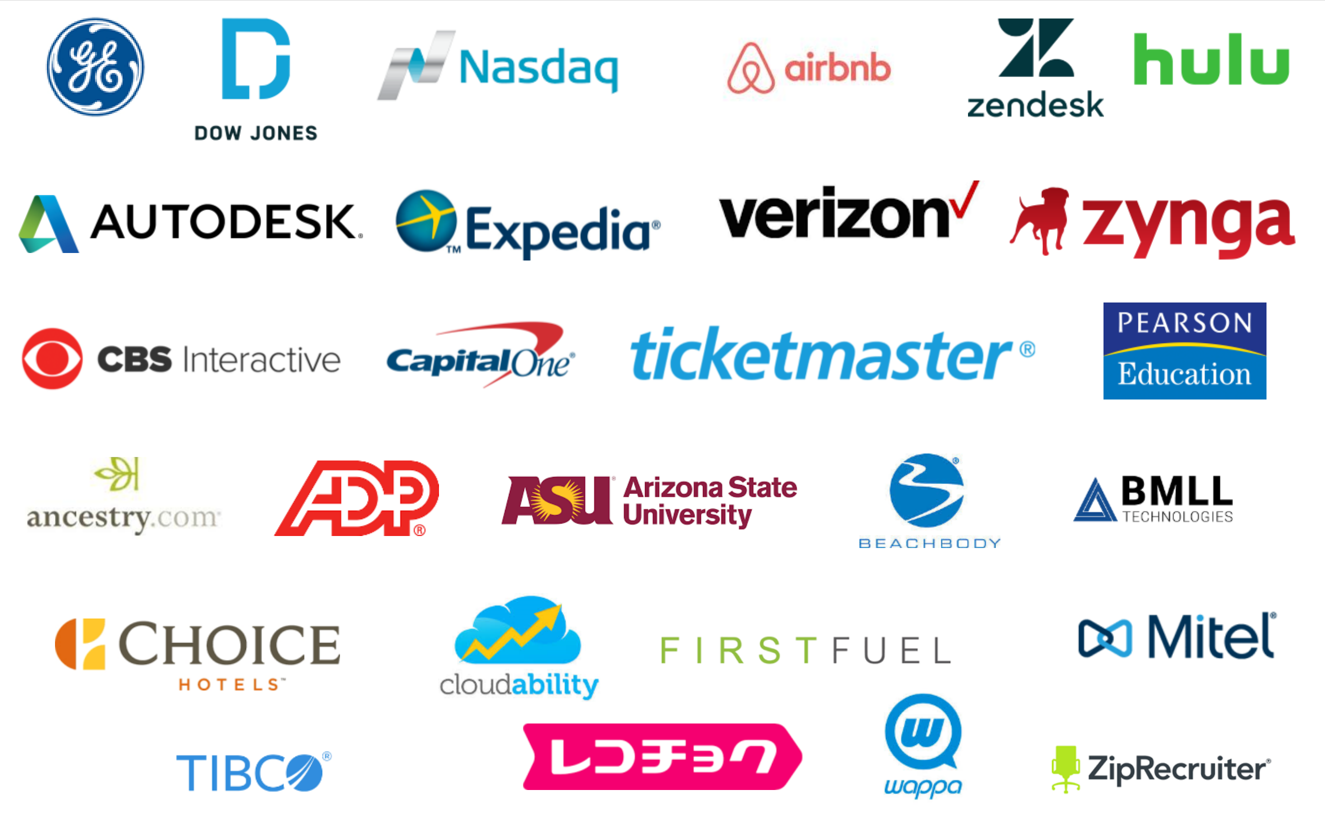
Routine maintenance

Takes care of your time-consuming database management tasks, freeing you to focus on your applications and business

Aurora customer adoption

Fastest growing service in AWS history

Aurora is used by ¾ of the top 100 AWS customers



Who is moving to Aurora and why?

Customers using open source engines

- Higher performance – up to 5x
- Better availability and durability
- Reduces cost – up to 60%
- Easy migration; no application change

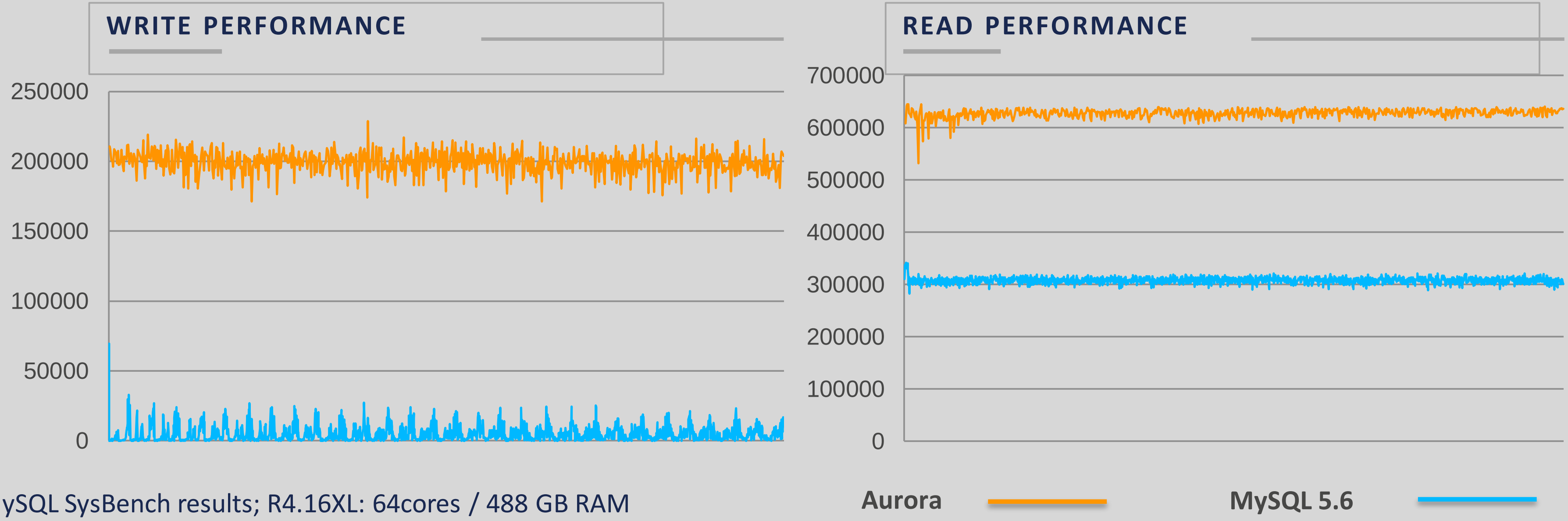
Customers using Commercial engines

- One tenth of the cost; no licenses
- Integration with cloud ecosystem
- Comparable performance and availability
- Migration tooling and services

AMAZON AURORA IS FAST…

5x faster than MySQL; 3x faster than PostgreSQL

Aurora MySQL performance

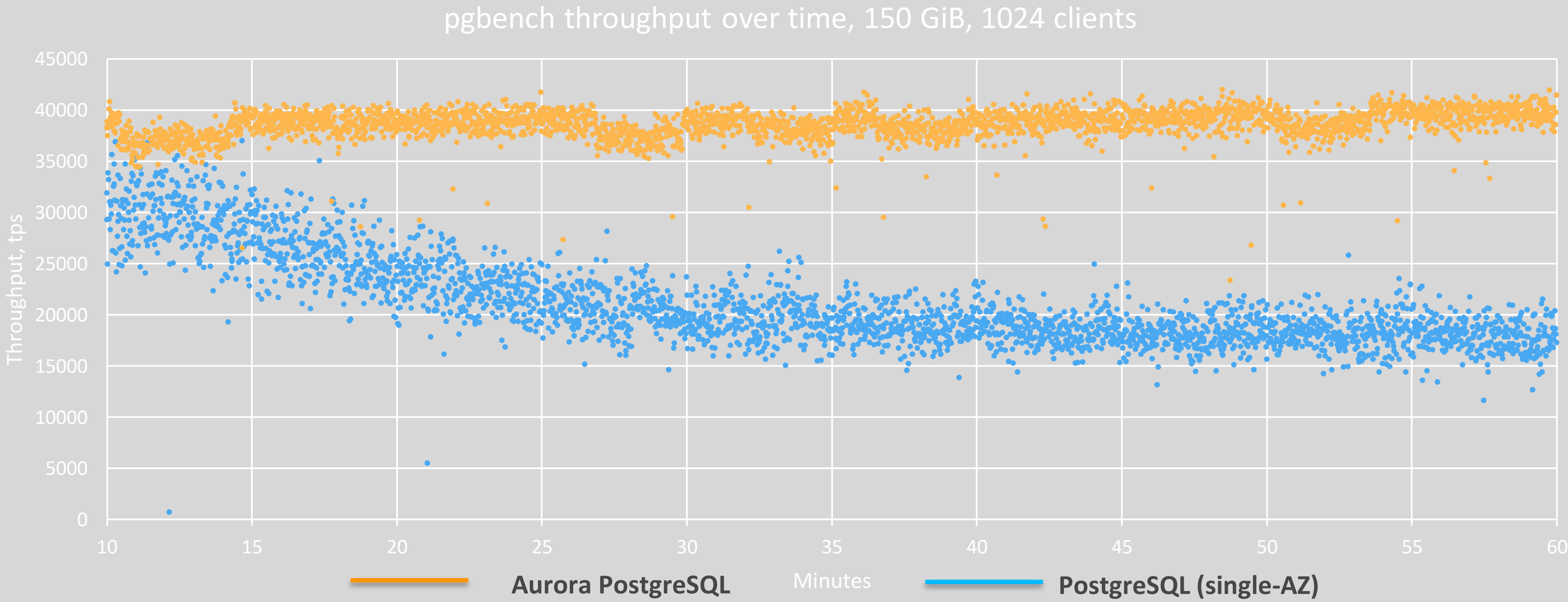


MySQL SysBench results; R4.16XL: 64cores / 488 GB RAM

Aurora read write throughput compared to MySQL 5.6 based on industry standard benchmarks.

Aurora PostgreSQL performance

While running pgbench at load, throughput is 3x more consistent than PostgreSQL



How did we achieve this?

DO LESS WORK

Do fewer IOs

Minimize network packets

Cache prior results

Offload the database engine

BE MORE EFFICIENT

Process asynchronously

Reduce latency path

Use lock-free data structures

Batch operations together

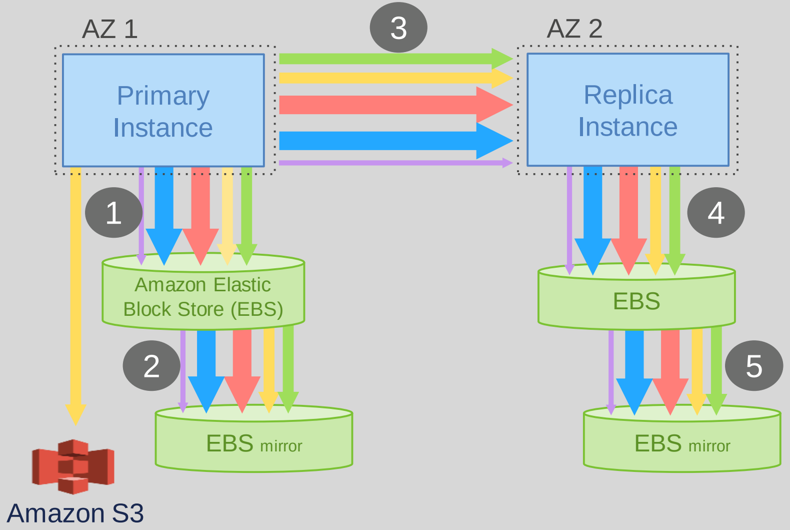
DATABASES ARE ALL ABOUT I/O

NETWORK-ATTACHED STORAGE IS ALL ABOUT PACKETS/SECOND

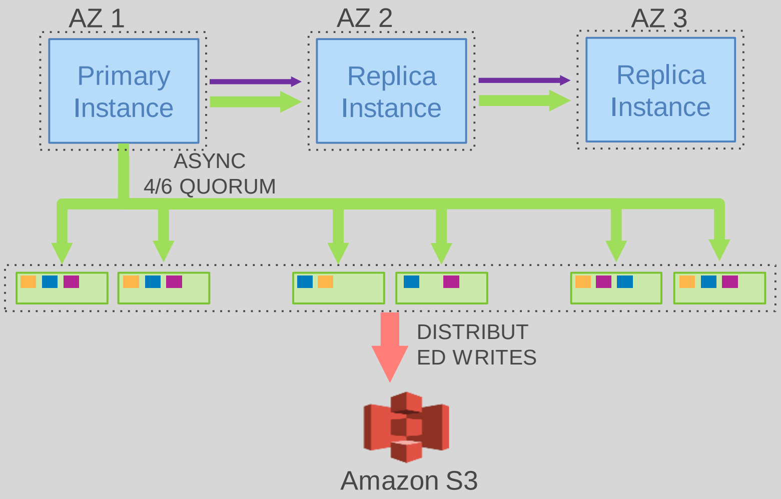
HIGH-THROUGHPUT PROCESSING IS ALL ABOUT CONTEXT SWITCHES

Aurora I/O profile

MYSQL WITH REPLICA



AMAZON AURORA



MySQL I/O profile for 30 min Sysbench run

780K transactions

7,388K I/Os per million txns (excludes mirroring, standby)

Average 7.4 I/Os per transaction

Aurora IO profile for 30 min Sysbench run

27,378K transactions **35X MORE**

0.95 I/Os per transaction (6X amplification) **7.7X LESS**

TYPE OF WRITE

LOG

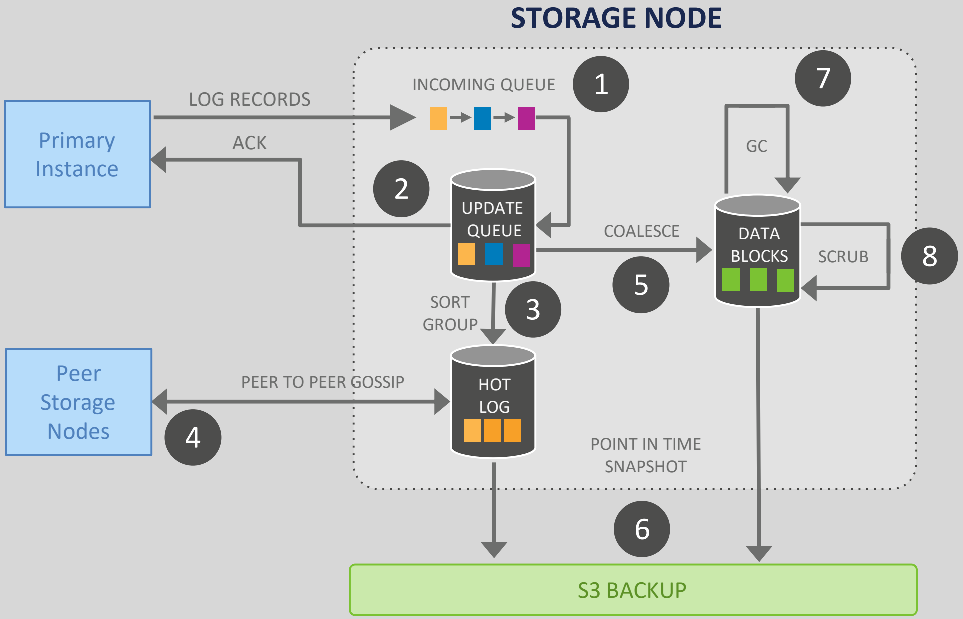
BINLOG

DATA

DOUBLE-WRITE

FRM FILES

IO traffic in Aurora (storage node)



IO FLOW

1. Receive record and add to in-memory queue
2. Persist record and ACK
3. Organize records and identify gaps in log
4. Gossip with peers to fill in holes
5. Coalesce log records into new data block versions
6. Periodically stage log and new block versions to S3
7. Periodically garbage collect old versions
8. Periodically validate CRC codes on blocks

OBSERVATIONS

All steps are asynchronous

Only steps 1 and 2 are in foreground latency path

Input queue is **46X less** than MySQL (unamplified, per node)

Favor latency-sensitive operations

Use disk space to buffer against spikes in activity

Performance enhancements in Aurora

DML throughput

- Adaptive thread pool
- Smart thread scheduler
- Asynchronous group commit
- Latch-free lock manager
- Lock compression
- Latch-free read views
- Fast B-Tree inserts
- Z-order spatial indexes
- Smart read-node selector
- Logical read ahead
- NUMA aware scheduler
- Highly concurrent catalog

Query execution

- Hash joins
- Batched scans
- Asynchronous key prefetch

DDL and Ops

- Instant schema update
- Faster index build
- High-performance auditing

What about availability?

“Performance only matters if your database is up”

6-way replicated storage

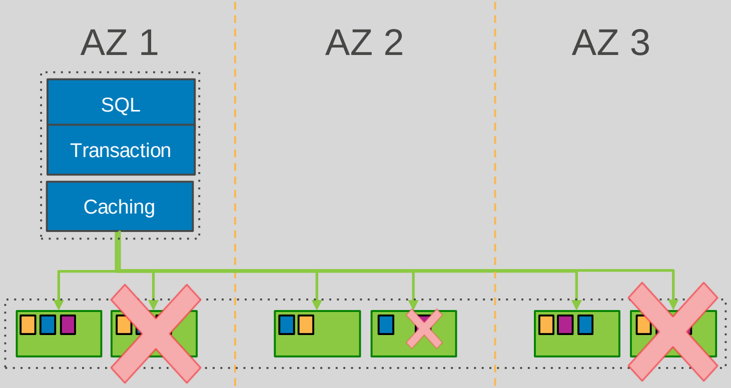
Survives catastrophic failures

Six copies across three availability zones

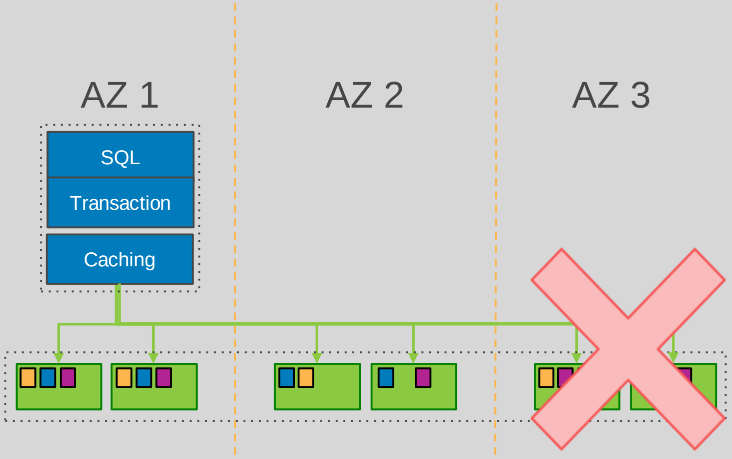
4 out 6 write quorum; 3 out of 6 read quorum

Peer-to-peer replication for repairs

Volume striped across hundreds of storage nodes

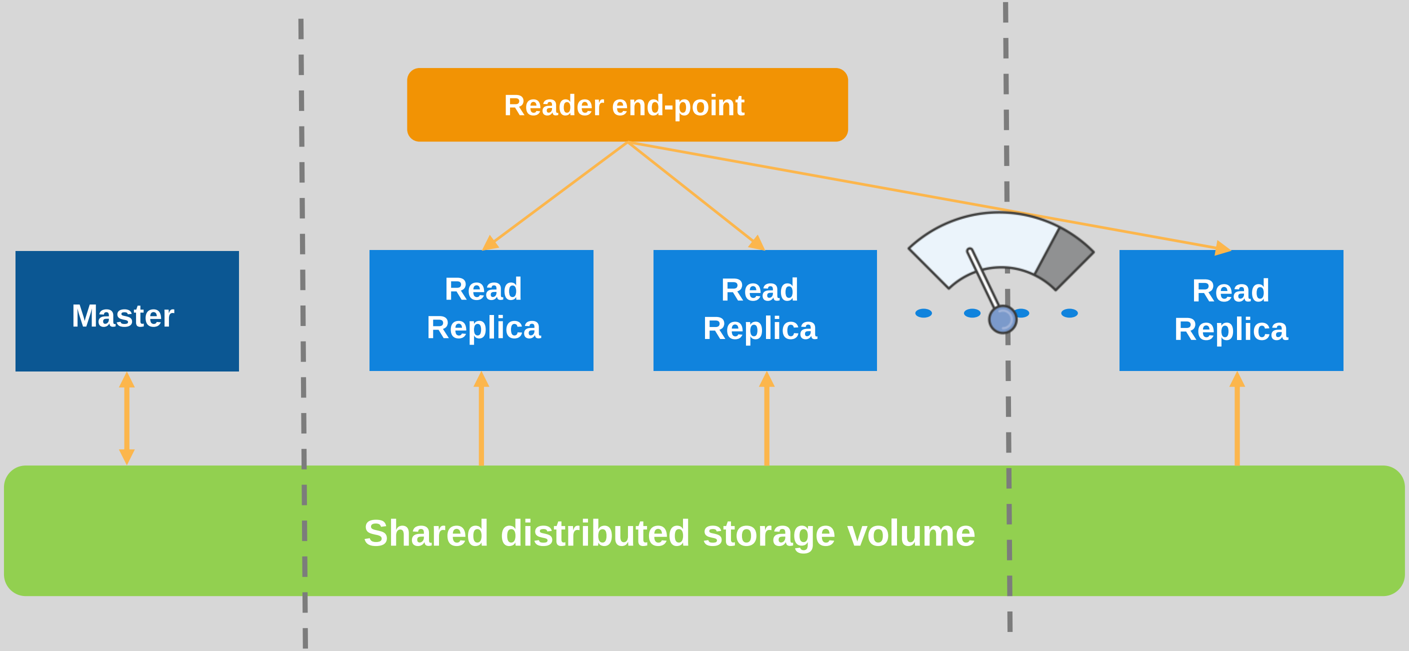


Read availability



Read and write availability

Up to 15 promotable read replicas



- ▶ Up to 15 promotable read replicas across multiple availability zones
- ▶ Re-do log based replication leads to low replica lag – typically < 10ms
- ▶ Reader end-point with load balancing and **auto-scaling**

Availability enhancements in Aurora

Unplanned unavailability

- Instant crash recovery
- Survivable cache
- Fast failover, incl. driver support

Planned unavailability

- Zero-downtime patching

Business continuity planning

- Continuous automated backup
- Point-in-Time-Restore
- Backtrack
- Cross-region read replicas

Instant crash recovery

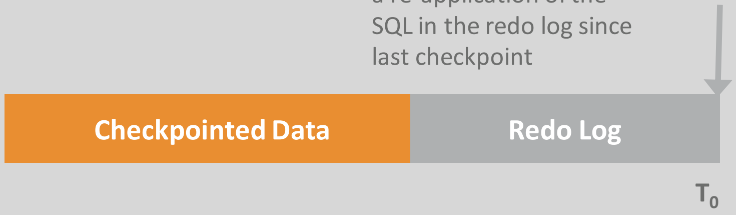
TRADITIONAL DATABASE

Have to replay logs since the last checkpoint

Typically 5 minutes between checkpoints

Single-threaded in MySQL; requires a large number of disk accesses

Crash at T_0 requires a re-application of the SQL in the redo log since last checkpoint



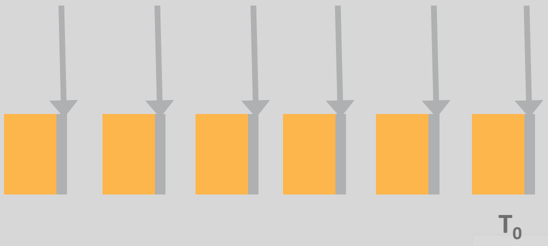
AMAZON AURORA

Underlying storage replays redo records on demand as part of a disk read

Parallel, distributed, asynchronous

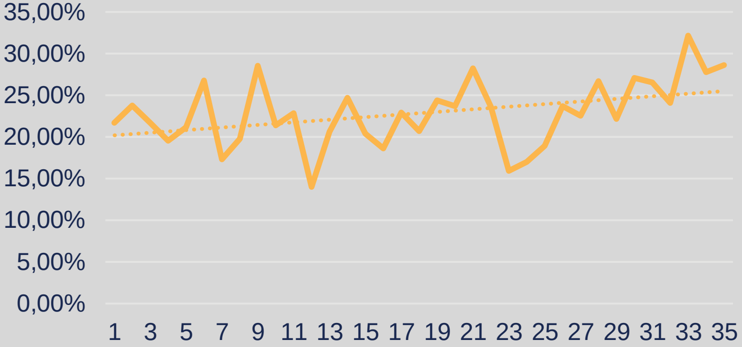
No replay for startup

Crash at T_0 will result in redo logs being applied to each segment on demand, in parallel, asynchronously

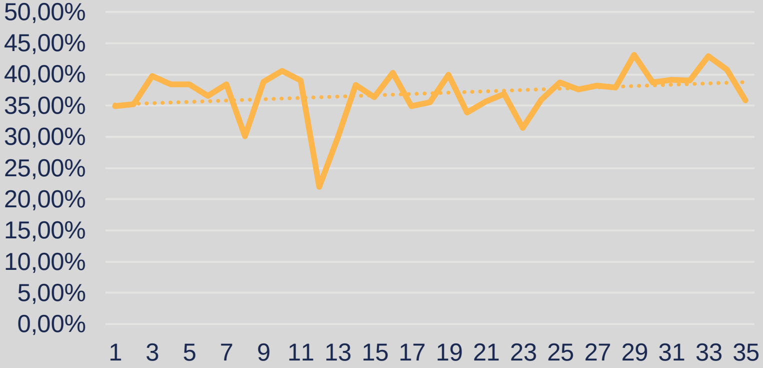


Database fail-over time

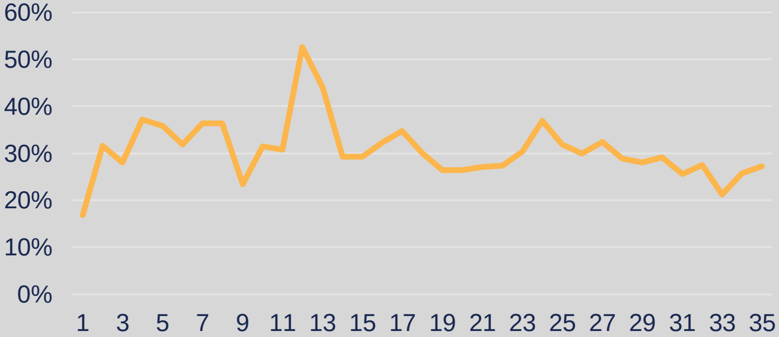
0 - 5s – 30% of fail-overs



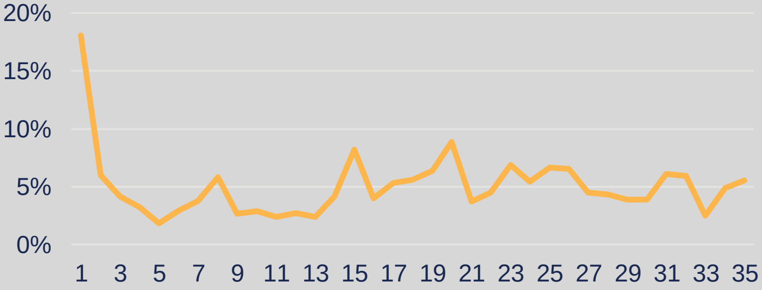
5 - 10s – 40% of fail-overs



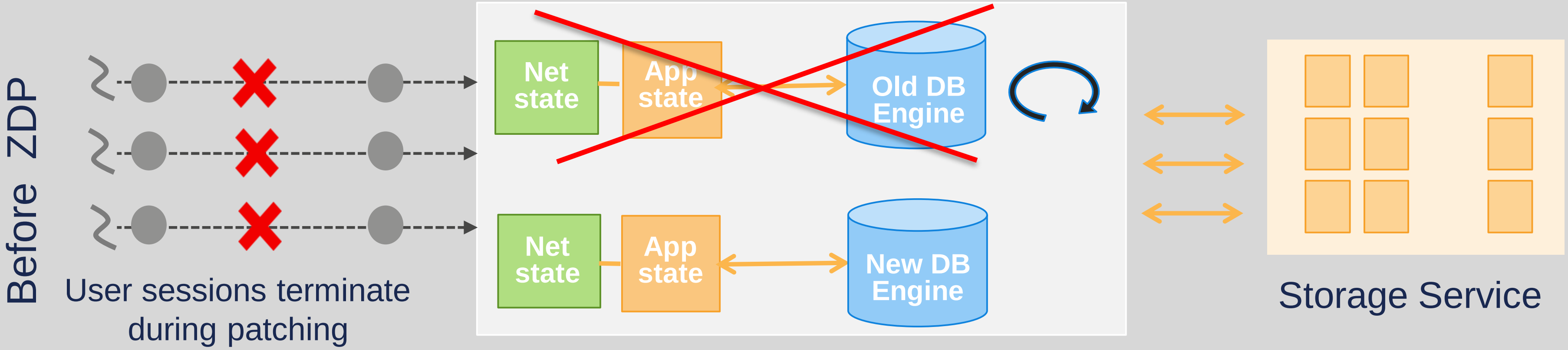
10 - 20s – 25% of fail-overs



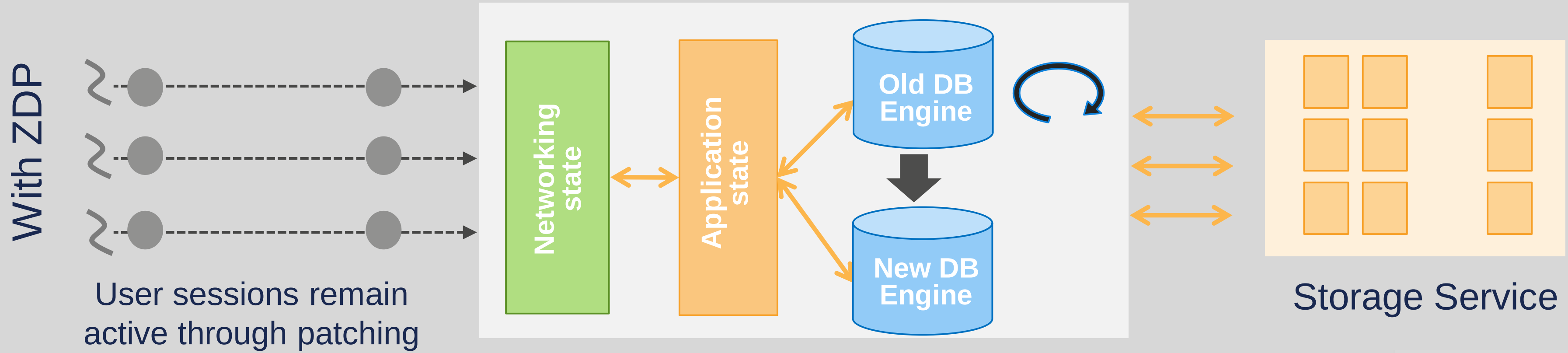
20 - 30s – 5% of fail-overs



Zero downtime patching



User sessions terminate during patching



User sessions remain active through patching

Recent Innovations

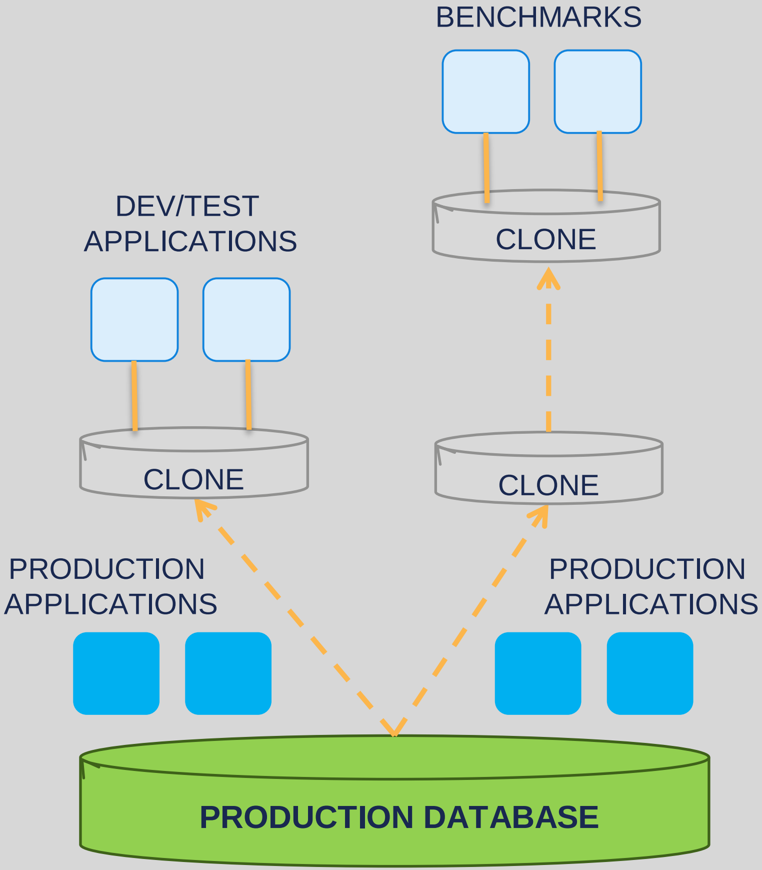
Fast database cloning

Clone database without copying data

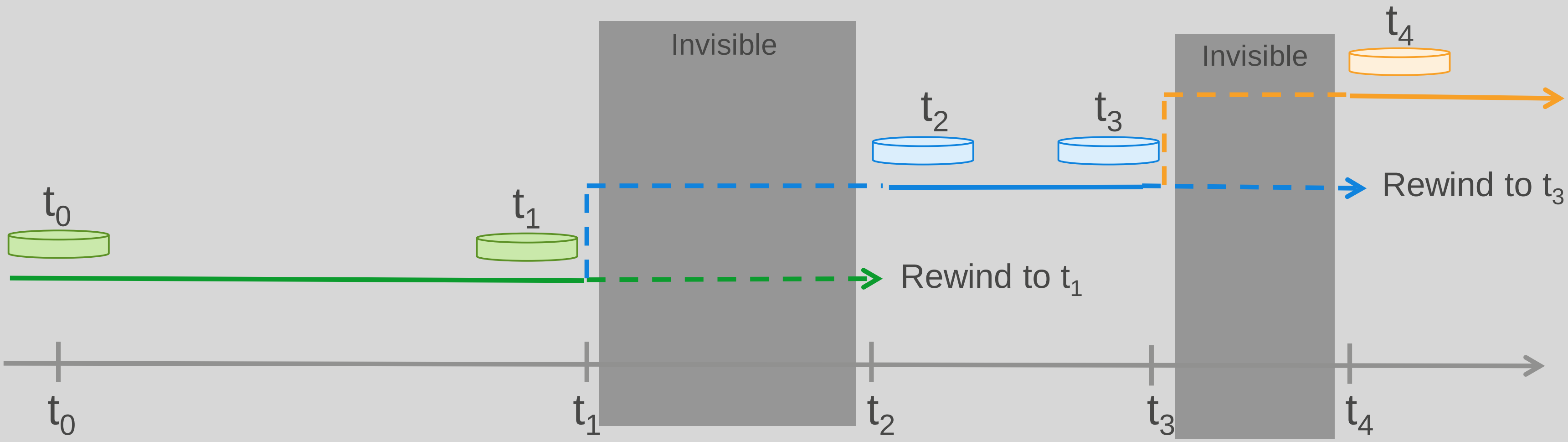
- Creation of a clone is nearly instantaneous
- Data copy happens only on write – when original and cloned volume data differ

Example use cases

- Clone a production DB to run tests
- Reorganize a database
- Save a point in time snapshot for analysis without impacting production system.



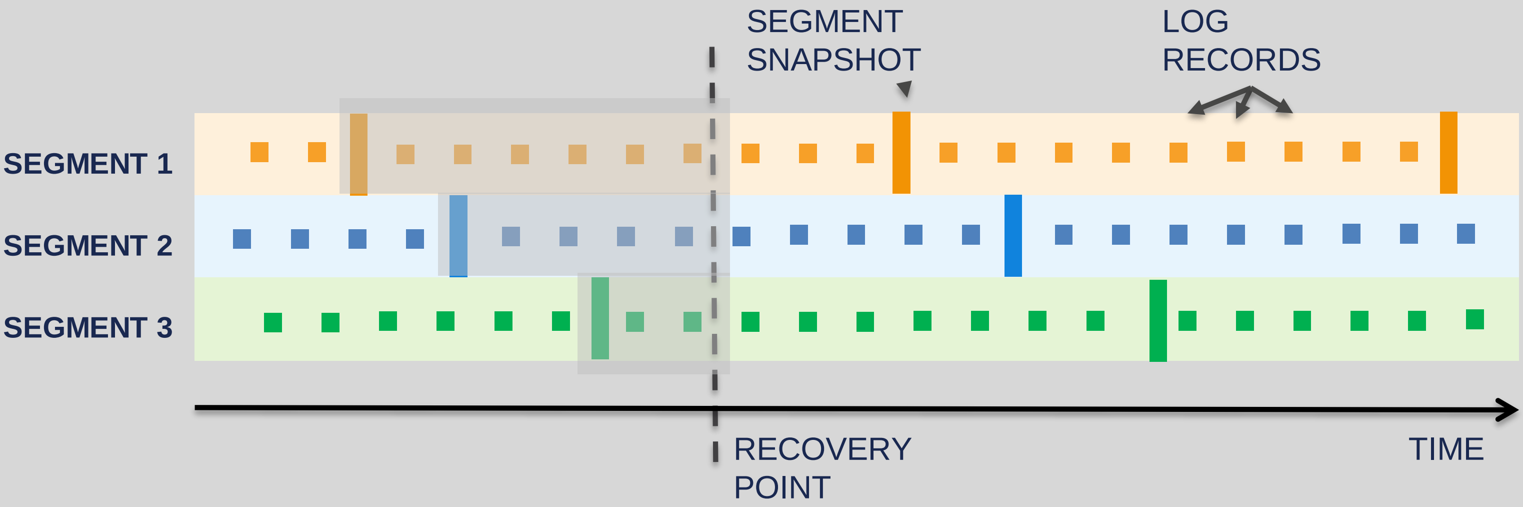
Database backtrack



Backtrack brings the database to a point in time without requiring restore from backups

- Backtracking from an unintentional DML or DDL operation
- Backtrack is not destructive. You can backtrack multiple times to find the right point in time

How does backtrack work?



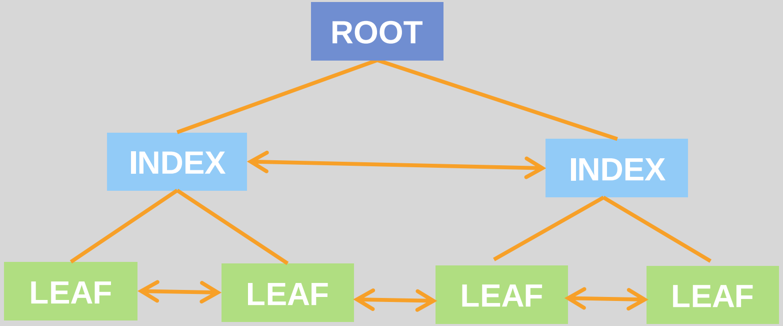
We keep periodic snapshot of each segment; we also preserve the redo logs

For backtrack, we identify the appropriate segment snapshots

Apply log streams to segment snapshots in parallel and asynchronously

Online DDL: MySQL vs. Aurora

MySQL



- Full Table copy in the background
- Rebuilds all indexes – can take hours or days
- DDL operation impacts DML throughput
- Table lock applied to apply DML changes

Aurora

table name	operation	column-name	time-stamp
Table 1	add-col	column-abc	t1
Table 2	add-col	column-qpr	t2
Table 3	add-col	column-xyz	t3

- Use schema versioning to decode the block.
- Modify-on-write primitive to upgrade to latest schema
- Currently support add NULLable column at end of table
- **Add column anywhere and with default coming soon.**

Online DDL performance

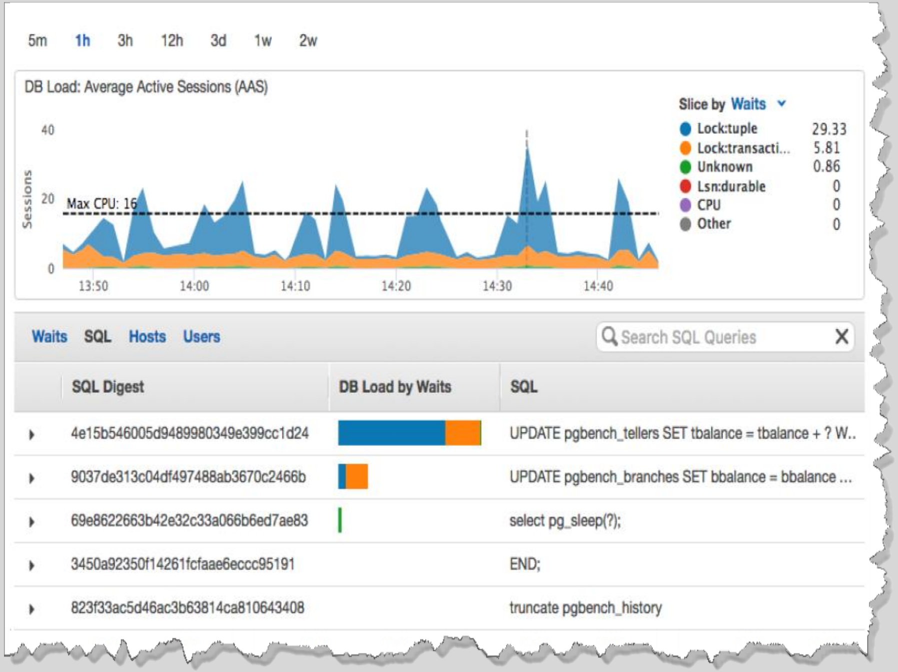
On r3.large

	Aurora	MySQL 5.6	MySQL 5.7
10GB table	0.27 sec	3,960 sec	1,600 sec
50GB table	0.25 sec	23,400 sec	5,040 sec
100GB table	0.26 sec	53,460 sec	9,720 sec

On r3.8xlarge

	Aurora	MySQL 5.6	MySQL 5.7
10GB table	0.06 sec	900 sec	1,080 sec
50GB table	0.08 sec	4,680 sec	5,040 sec
100GB table	0.15 sec	14,400 sec	9,720 sec

Performance Insights



Slice by Waits	
● Lock:tuple	29.33
● Lock:transacti...	5.81
● Unknown	0.86
● Lsn:durable	0
● CPU	0
● Other	0

SQL Digest	DB Load by Waits	SQL
4e15b546005d9489980349e399cc1d24		UPDATE pgbench_tellers SET tbalance = tbalance + ? W..
9037de313c04df497488ab3670c2466b		UPDATE pgbench_branches SET bbalance = bbalance ...
69e8622663b42e32c33a066b6ed7ae83		select pg_sleep(?);
3450a92350f14261fcfaae6eccc95191		END;
823f33ac5d46ac3b63814ca810643408		truncate pgbench_history

Dashboard showing load on DB

- Easy
- Powerful

Identifies source of bottlenecks

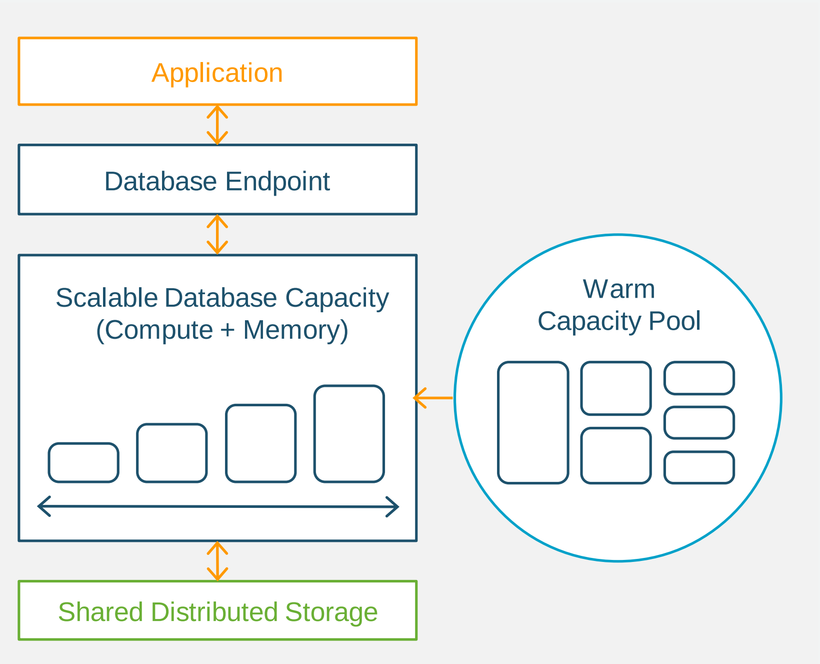
- Top SQL

Adjustable time frame

- Hour, day, week , month
- Up to 35 days of data

Aurora Serverless

On-demand, auto-scaling database for applications with variable workloads


Application
Database Endpoint
Scalable Database Capacity (Compute + Memory)
Warm Capacity Pool
Shared Distributed Storage

Starts up on demand, shuts down when not in use

Automatically scales with no instances to manage

Pay per second for the database capacity you use

Aurora Parallel Query

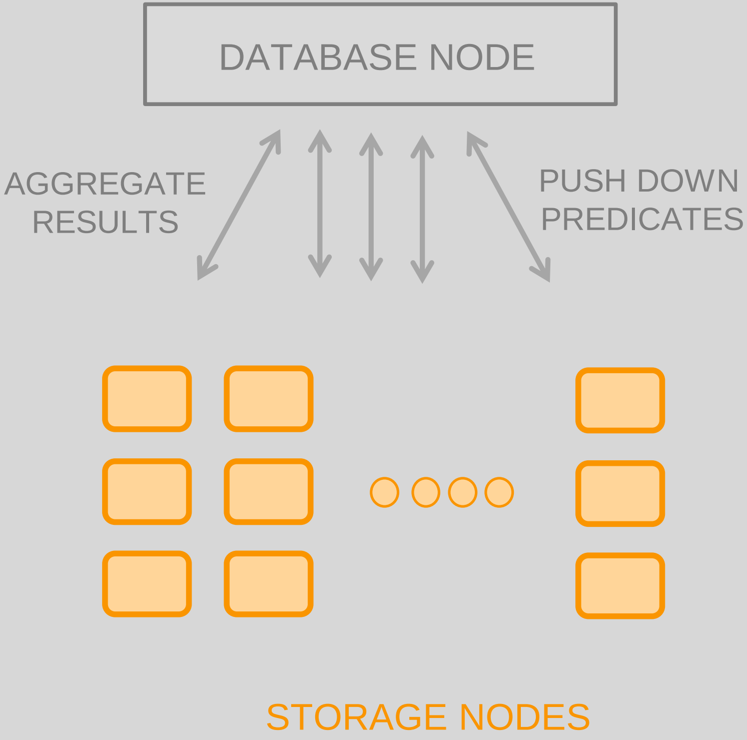
Parallel query processing

Aurora storage has thousands of CPUs

- Presents opportunity to push down and parallelize query processing using the storage fleet
- Moving processing close to data reduces network traffic and latency

However, there are significant challenges

- Data stored in storage node is not range partitioned – require full scans
- Data may be in-flight
- Read views may not allow viewing most recent data
- Not all functions can be pushed down to storage nodes



Parallel Query: Use cases

Orders of magnitude faster queries

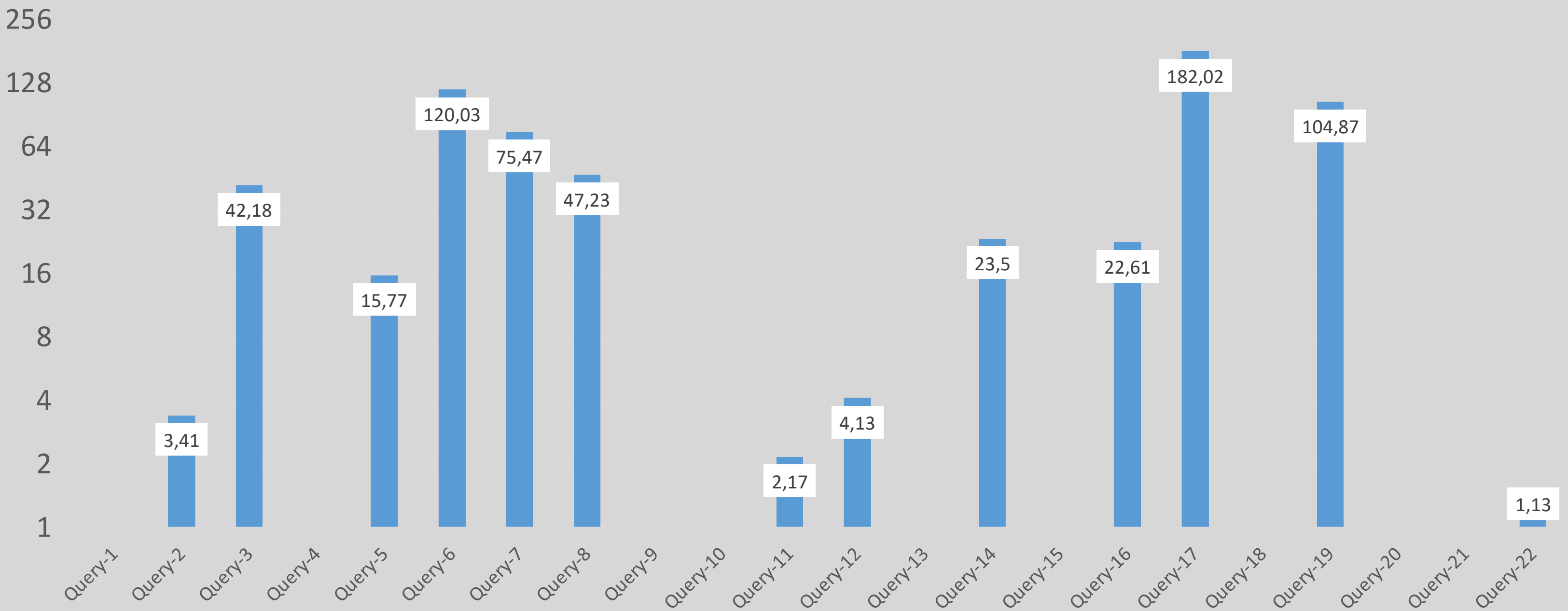
Parallelism increases with data size

Reduced contention with OLTP workload

Analytic workloads on OLTP working set

- Analyze real-time data
- Avoid building ETL pipelines for ad-hoc queries
- A large number of concurrent analytic queries

Parallel query performance



“We noticed query time reduce from 32 minutes to 3 minutes.” – preview customer (online media company)

Processing at head node

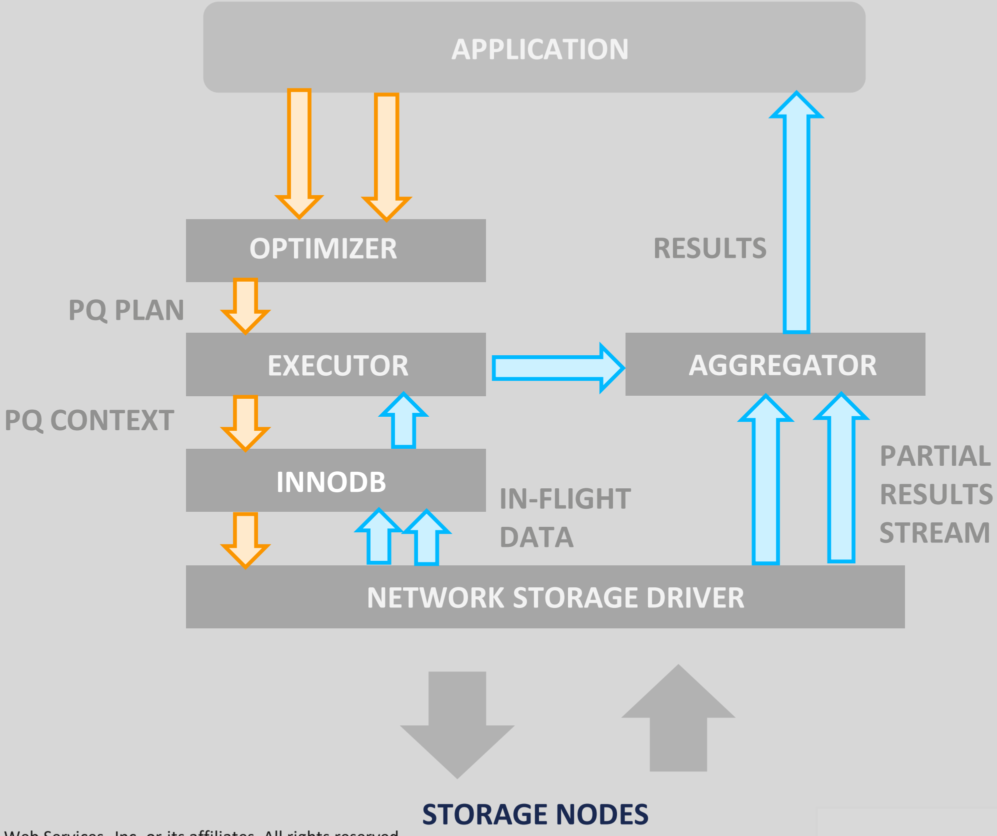
Query Optimizer produces PQ Plan and creates PQ context based on leaf page discovery

PQ request is sent to storage node along with PQ context

Storage node produces

- Partial results streams with processed stable rows
- Raw stream of unprocessed rows with pending undos

Head node aggregates these data streams to produce final results



Processing at storage node

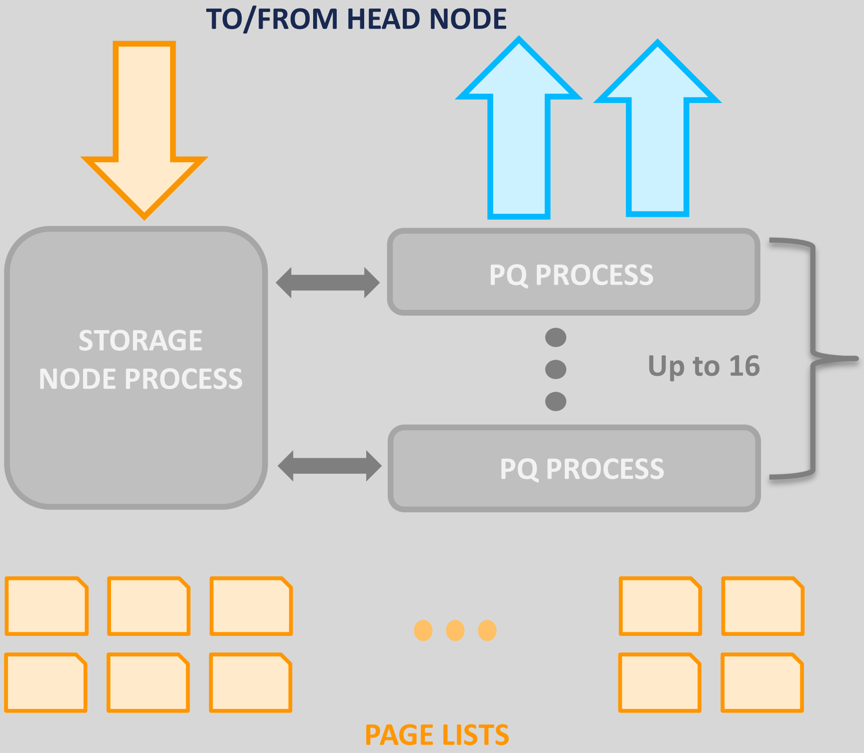
Each storage node runs up to 16 PQ processes, each associated with a parallel query

PQ process receives PQ context

- List of pages to scan
- Read view and projections
- Expression evaluation byte code

PQ process makes two passes through the page list

- **Pass 1:** Filter evaluation on InnoDB formatted raw data
- **Pass 2:** Expression evaluation on MySQL formatted data



What is Amazon Aurora?

Database reimagined for the cloud

☑ **Speed** and **availability** of high-end commercial databases

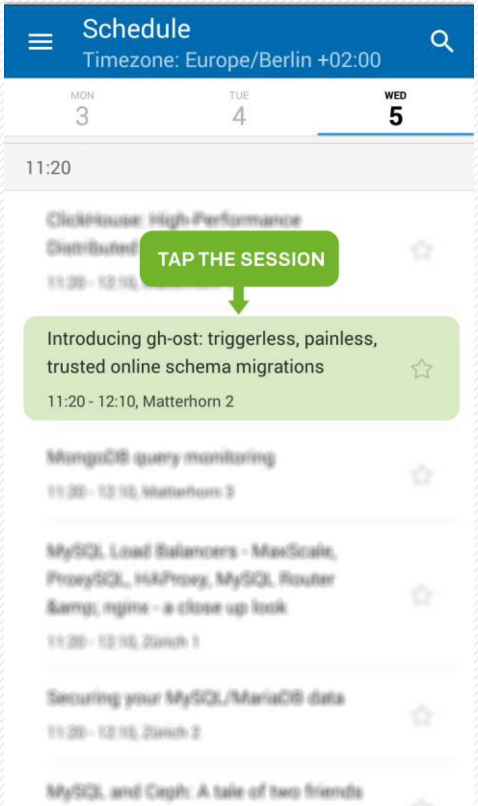
☑ **Simplicity** and **cost-effectiveness** of open source databases

☑ Drop-in **compatibility** with MySQL and PostgreSQL

☑ Simple **pay as you go** pricing

Delivered as a **managed** service

Rate My Session



Schedule
Timezone: Europe/Berlin +02:00

11:20

ClickHouse: High Performance Distributed...
11:20 - 12:10, ...

TAP THE SESSION

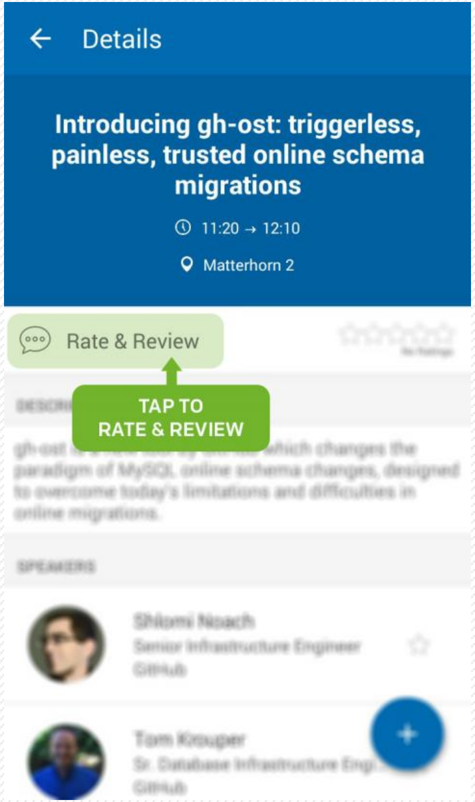
Introducing gh-ost: triggerless, painless, trusted online schema migrations
11:20 - 12:10, Matterhorn 2

MongoDB query monitoring
11:20 - 12:10, Matterhorn 3

MySQL Load Balancers - MaxScale, ProxySQL, HAProxy, MySQL Router &amp; nginx - a close up look
11:20 - 12:10, Zürich 1

Securing your MySQL/MariaDB data
11:20 - 12:10, Zürich 2

MySQL and Ceph: A tale of two friends



Details

Introducing gh-ost: triggerless, painless, trusted online schema migrations

11:20 → 12:10

Matterhorn 2

Rate & Review

TAP TO RATE & REVIEW

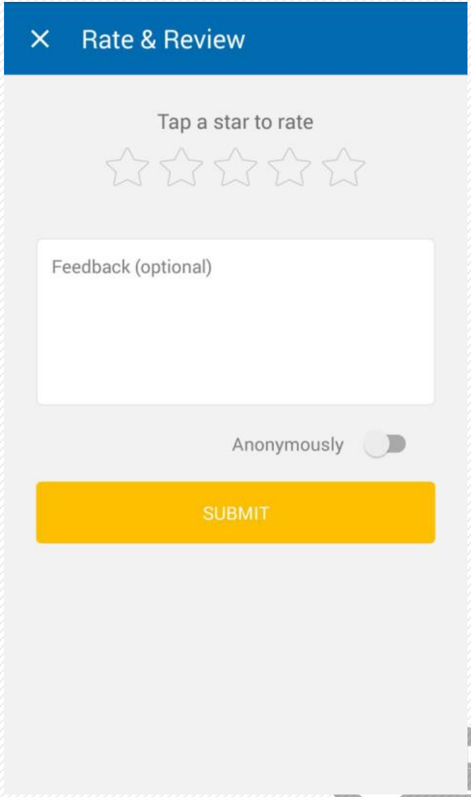
DESCRIPTION

gh-ost is a new tool by GitHub which changes the paradigm of MySQL online schema changes, designed to overcome today's limitations and difficulties in online migrations.

SPEAKERS

Shlomi Noach
Senior Infrastructure Engineer
GitHub

Tom Krouper
Sr. Database Infrastructure Engi...
GitHub



Rate & Review

Tap a star to rate

☆ ☆ ☆ ☆ ☆

Feedback (optional)

Anonymously

SUBMIT