DTCC

2017第八届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2017

让Amazon Aurora助您的业务腾飞

杨海俊

亚马逊 AWS 解决方案架构师

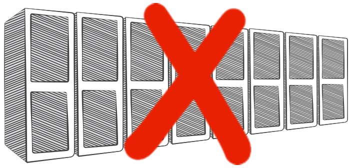
议题

- ▶ AWS 和 Amazon RDS 概览
- ▶ 什么是 Amazon Aurora
- ▶ 为什么使用 Amazon Aurora
  - ▪ 速度更快，高可用性，易于使用，低拥有成本

AWS 云平台的特性

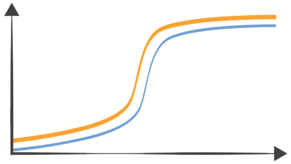
充分发挥AWS 云平台的特性

无前期投入成本
按需使用

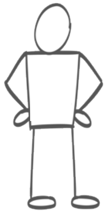


可持续的，
更低的TCO

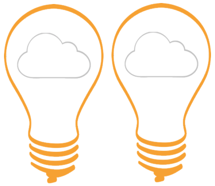
高扩展性
高弹性



灵敏运维
研发创新

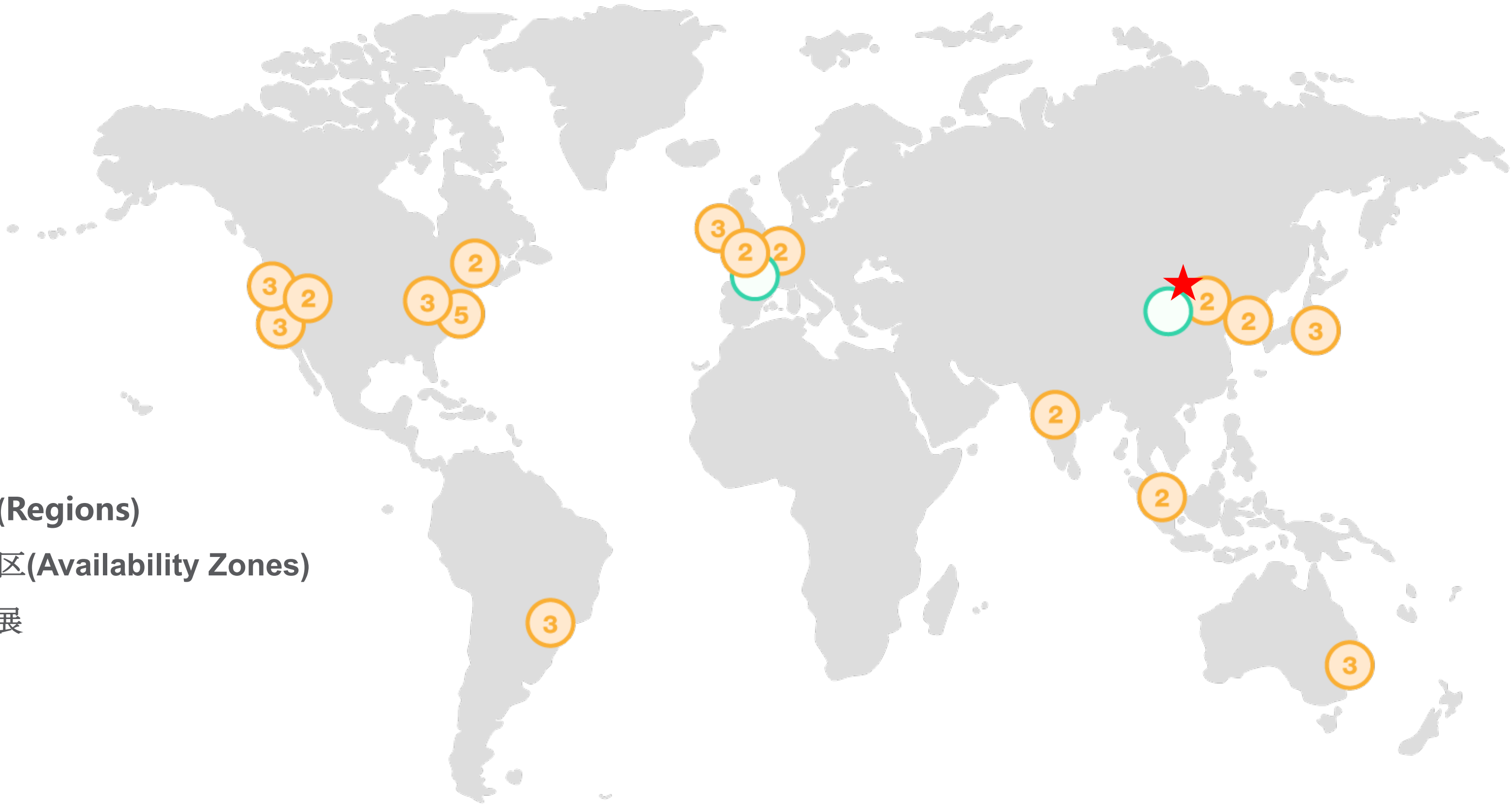


让您专注于
您的核心业务



跨区域的
全球化部署

AWS 全球基础设施

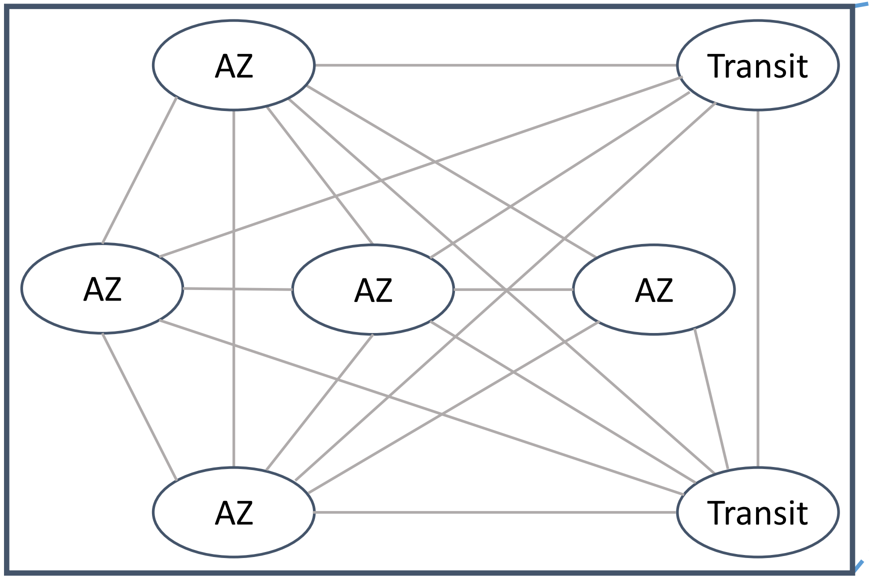


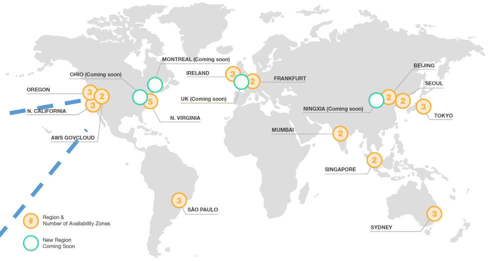
- **16**个区域**(Regions)**
- **42**个可用区**(Availability Zones)**
- 持续的扩展

AWS Global Infrastructure

https://aws.amazon.com/about-aws/global-infrastructure/?nc1=h_ls

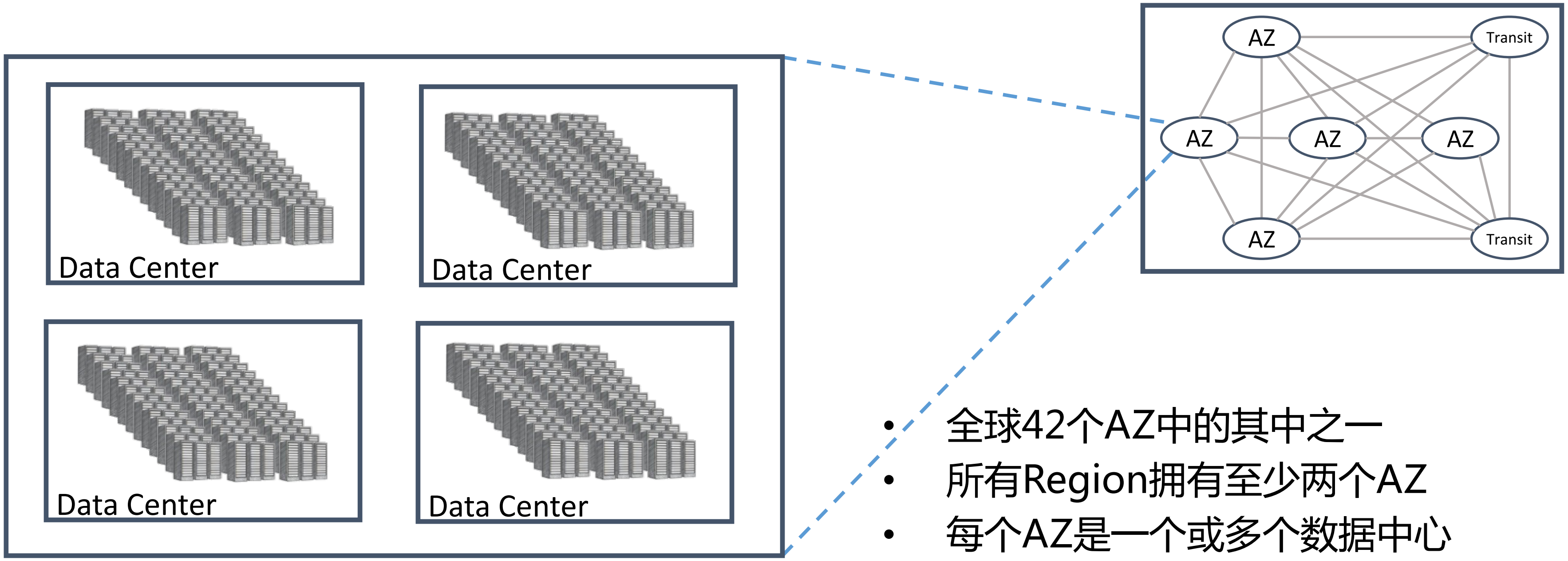
AWS Region 示例





- AWS全球16个region 之一
- 冗余的链路连接到转接中心
- 转接中心:
  - 连接至其他AWS Region的专有链路
  - 连接至Direct Connect客户的专有链路
  - 通过对等互联和付费转接连接至互联网
- AZ之间是城际DWDM链路
- Region内82864条光纤
- AZ间的延迟低于2ms，通常小于1ms
- AZ间的峰值流量可达25Tbps

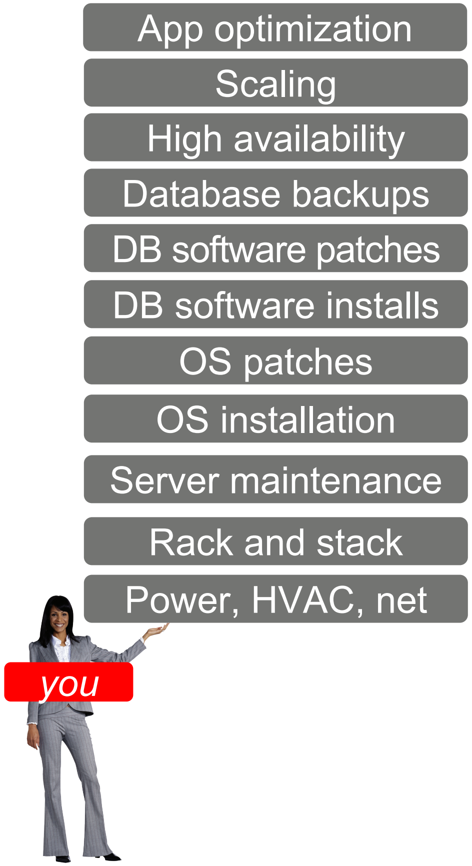
AWS 可用区示例



- 全球42个AZ中的其中之一
- 所有Region拥有至少两个AZ
- 每个AZ是一个或多个数据中心
  - 数据中心不跨AZ
  - 某些AZ有多达6个数据中心
- 在AZ内的数据中心间的延迟低于¼ ms
  - 无需跨AZ的独立性
  - 需要低延迟与全带宽

数据库自建还是托管

在自有数据中心搭建数据库服务



App optimization

Scaling

High availability

Database backups

DB software patches

DB software installs

OS patches

OS installation

Server maintenance

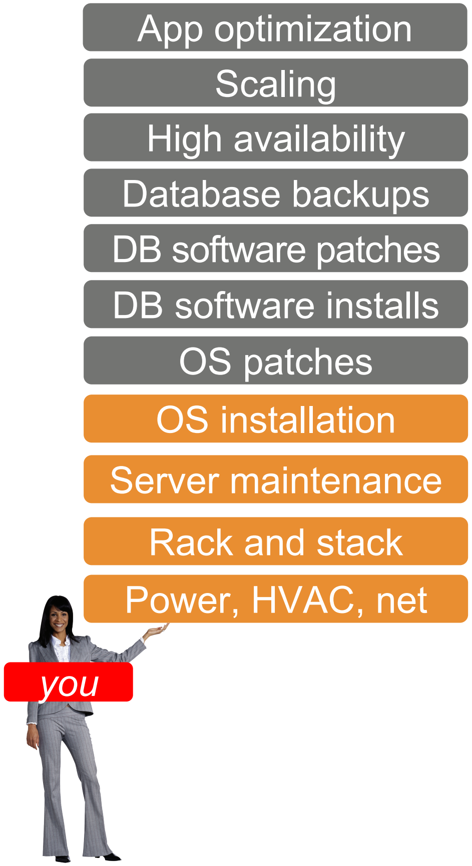
Rack and stack

Power, HVAC, net

you

数据库自建还是托管

在自有数据中心托管数据库服务



App optimization

Scaling

High availability

Database backups

DB software patches

DB software installs

OS patches

OS installation

Server maintenance

Rack and stack

Power, HVAC, net

you

数据库自建还是托管

基于AWS EC2构建数据库服务

- App optimization
- Scaling
- High availability
- Database backups
- DB software patches
- DB software installs
- OS patches

- OS installation
- Server maintenance
- Rack and stack
- Power, HVAC, net

amazon web services

数据库自建还是托管

如果选择托管的数据库服务

App optimization

Scaling

High availability

Database backups

DB software patches

DB software installs

OS patches

OS installation

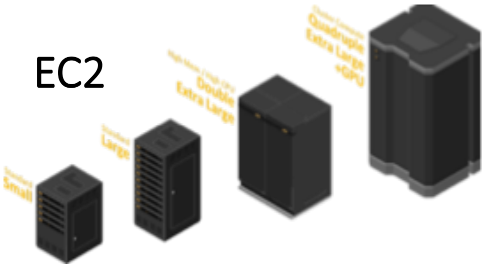
Server maintenance

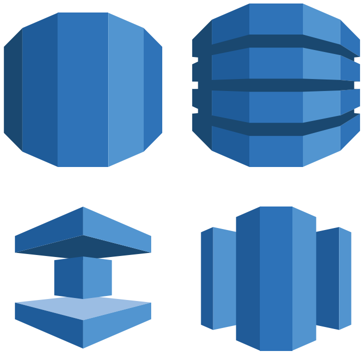
Rack and stack

Power, HVAC, net

amazon
webservices

数据库: 自建还是托管





自搭自建

- 通过EC2 instance 全面管理

  (Raid + 预制IOPS)
- 肩负数据库管理的所有重任：升级，备份，故障转移… …
- 全面负责数据库安全的方方面面
- 复杂的主备设定，副本管理以及数据管理

托管服务

- 从底层设施以及基础关机管理的任务中脱离出来
- 通过API 调用实现数据库生命周期管理的自动化
- 关注于数据库访问设定以及应用安全
- 轻松管理主从，副本

数据库服务：Amazon RDS

与现有应用兼容，可选数据库引擎

Amazon Aurora, MySQL, PostgreSQL, Oracle, SQL Server, MariaDB

点几下鼠标或者调用API，就可完成部署

- AWS负责patching, backups, replication
- 非常容易scale up

快速、可预测的数据库性能

根据需求确定IO吞吐量和存储卷大小

SQL Server：20,000 IOPS, 4TB; 其他30,000 IOPS, 6TB存储；

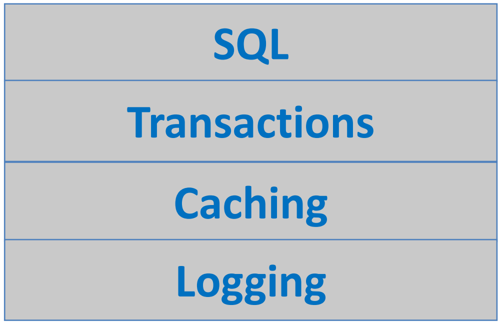
无固定资产投资，按使用付费

议题

- AWS 和 Amazon RDS 概览
- 什么是 Amazon Aurora
- 为什么使用 Amazon Aurora
    - 速度更快，高可用性，易于使用，低拥有成本

Amazon RDS for Aurora

- 为cloud重新设计的关系型数据库
- 企业级数据库
  - 提供商用数据库级别的性能和可用性，价格仅为1/10
  - 提供MySQL五倍的性能，与MySQL 5.6兼容
- 只为实际使用的存储付费
- 提供静态和传输中数据加密



SQL

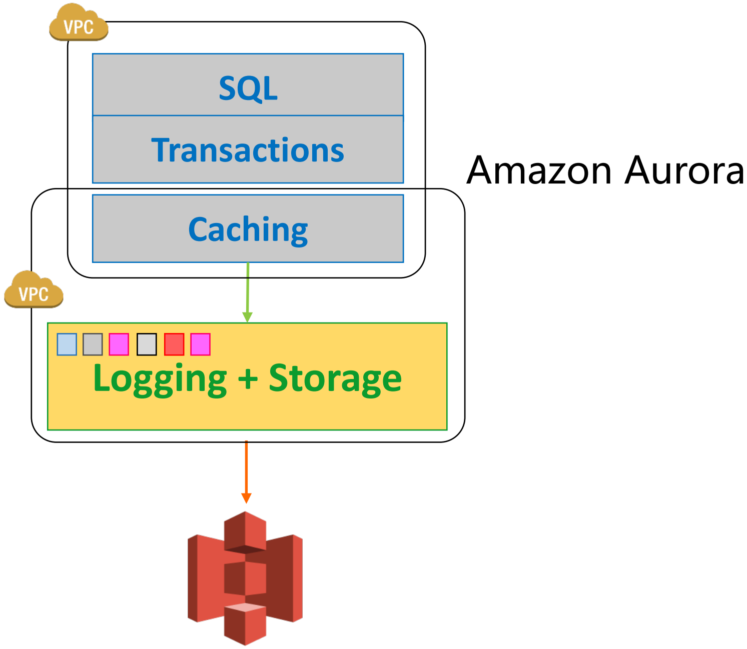
Transactions

Caching

Logging

传统数据库系统

Multiple layers of functionality
all in a monolithic stack



VPC

SQL

Transactions

Caching

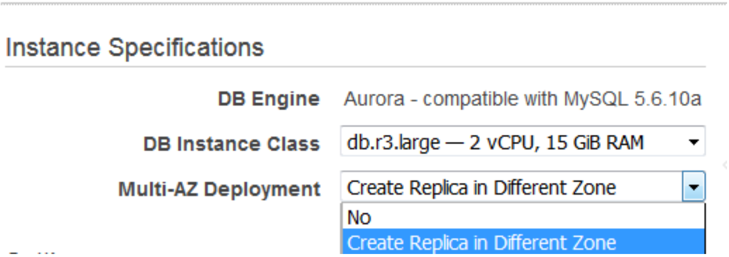
VPC

Logging + Storage

Amazon Aurora

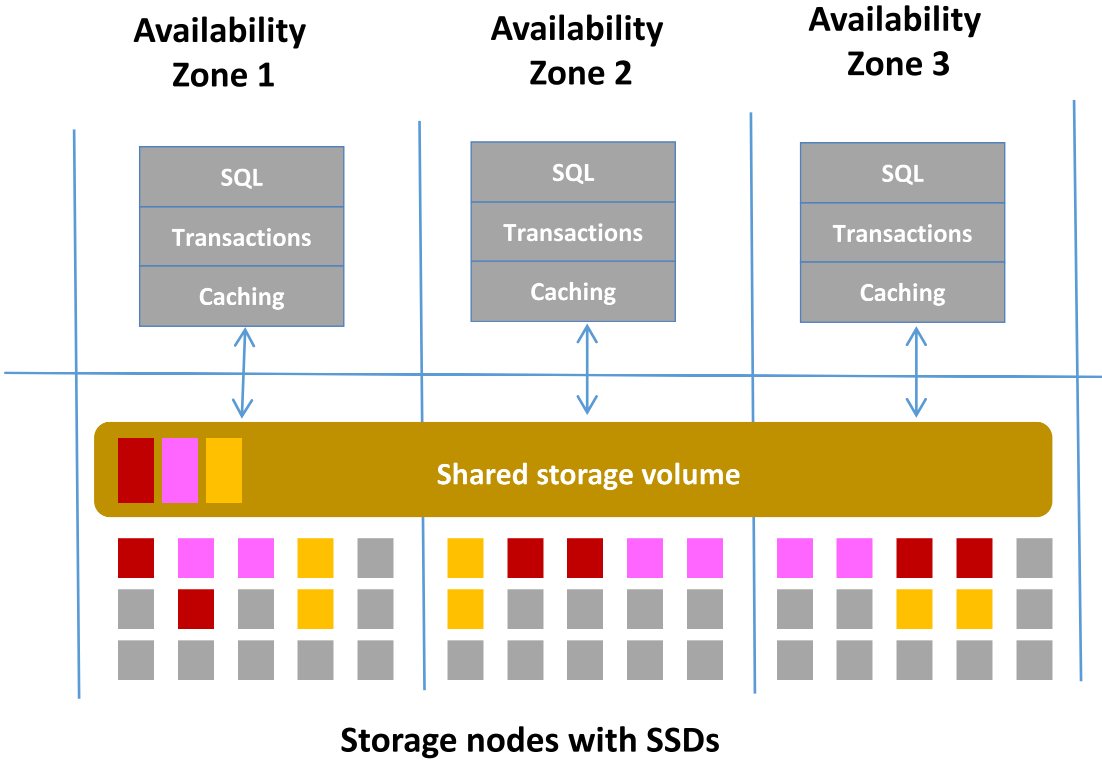
Amazon Aurora : 高扩展性, 分布式, 多租户架构

- 数据自动复制到3个AZ的6个存储节点
- 存储从10GB开始按用量增长，最大支持64TB
- 支持最多15 Replicas，可以作为故障转移目标



Instance Specifications

DB Engine	Aurora - compatible with MySQL 5.6.10a
DB Instance Class	db.r3.large — 2 vCPU, 15 GiB RAM
Multi-AZ Deployment	Create Replica in Different Zone



与其它 AWS 云平台服务无缝集成

Lambda Invoke Lambda events from stored procedures/triggers.

S3 Load data from S3, store snapshots and backups in S3.

IAM Use IAM roles to manage database access control.

CloudWatch Upload systems metrics and audit logs to CloudWatch.

议题

- ▶ AWS 和 Amazon RDS 概览
- ▶ 什么是 Amazon Aurora
- ▶ 为什么使用 Amazon Aurora
  - 速度更快，高可用性，易于使用，低拥有成本

完美契合企业级数据库系统的需求

企业级别的高可用性要求

- 跨3个可用区的 6-路 复制
- 30 秒内完成故障转移
- 快速的 crash recovery

性能和扩展性

- 高达 500 K/sec 读处理和 100 K/sec 写处理
- 15 个低延迟 (10 ms) Read Replicas
- 高达 64 TB 数据库优化存储卷

完全托管的云服务

- 快速 provisioning 和部署
- 自动安装补丁和软件升级
- 备份和 point-in-time 恢复
- 计算和存储的扩展性支持

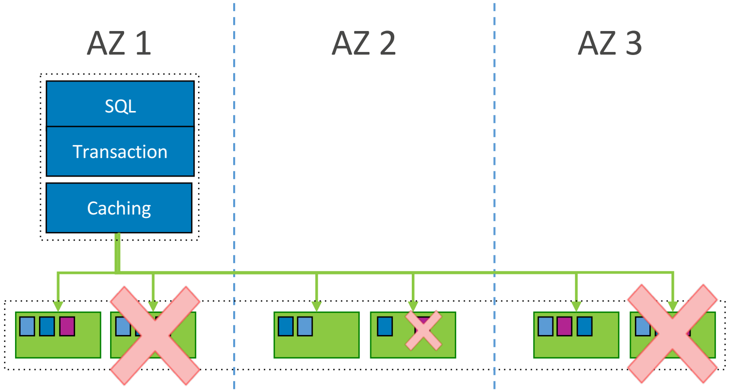
Amazon Aurora 高可用性

“Performance only matters if your database is up”

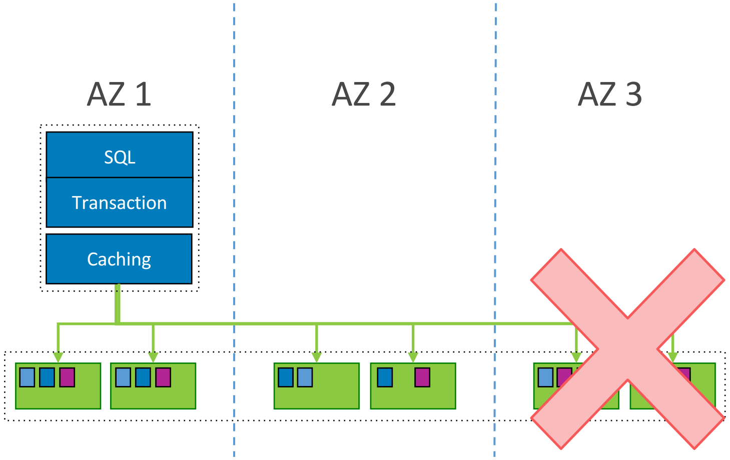
跨3个可用区的6路可复制存储

解决灾难性故障问题

- 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

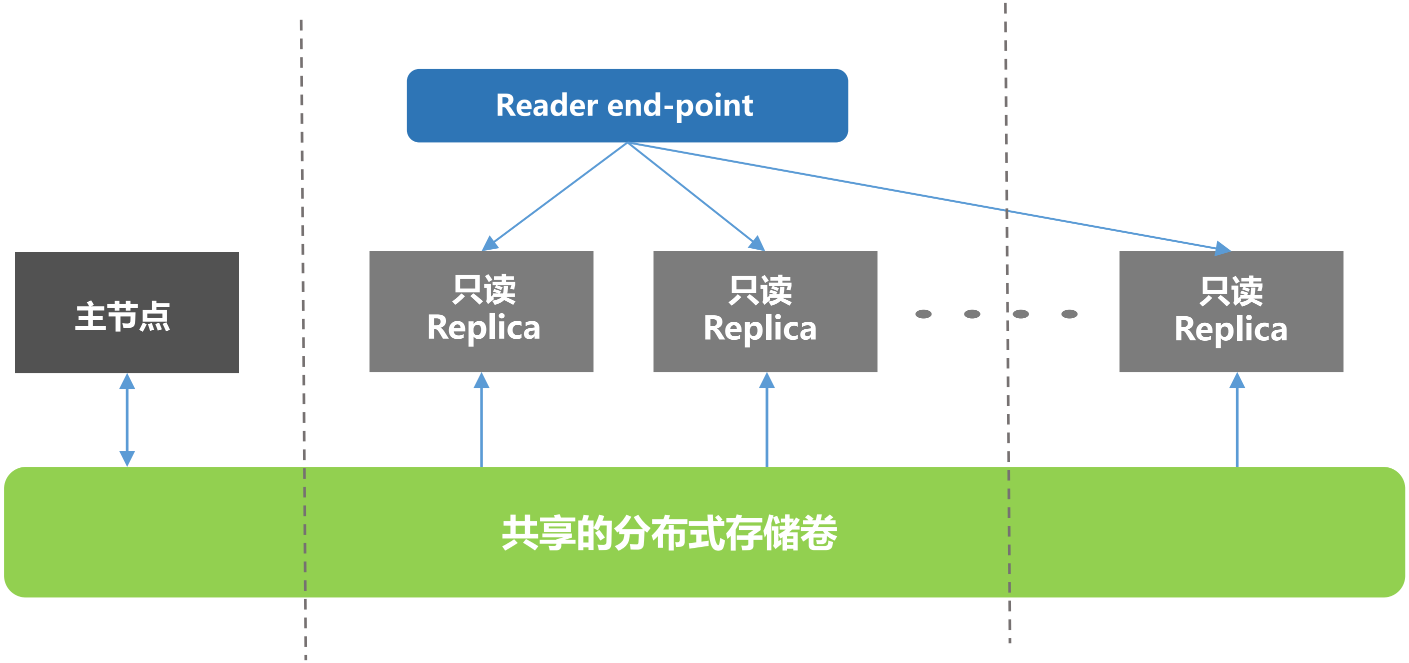


Read availability



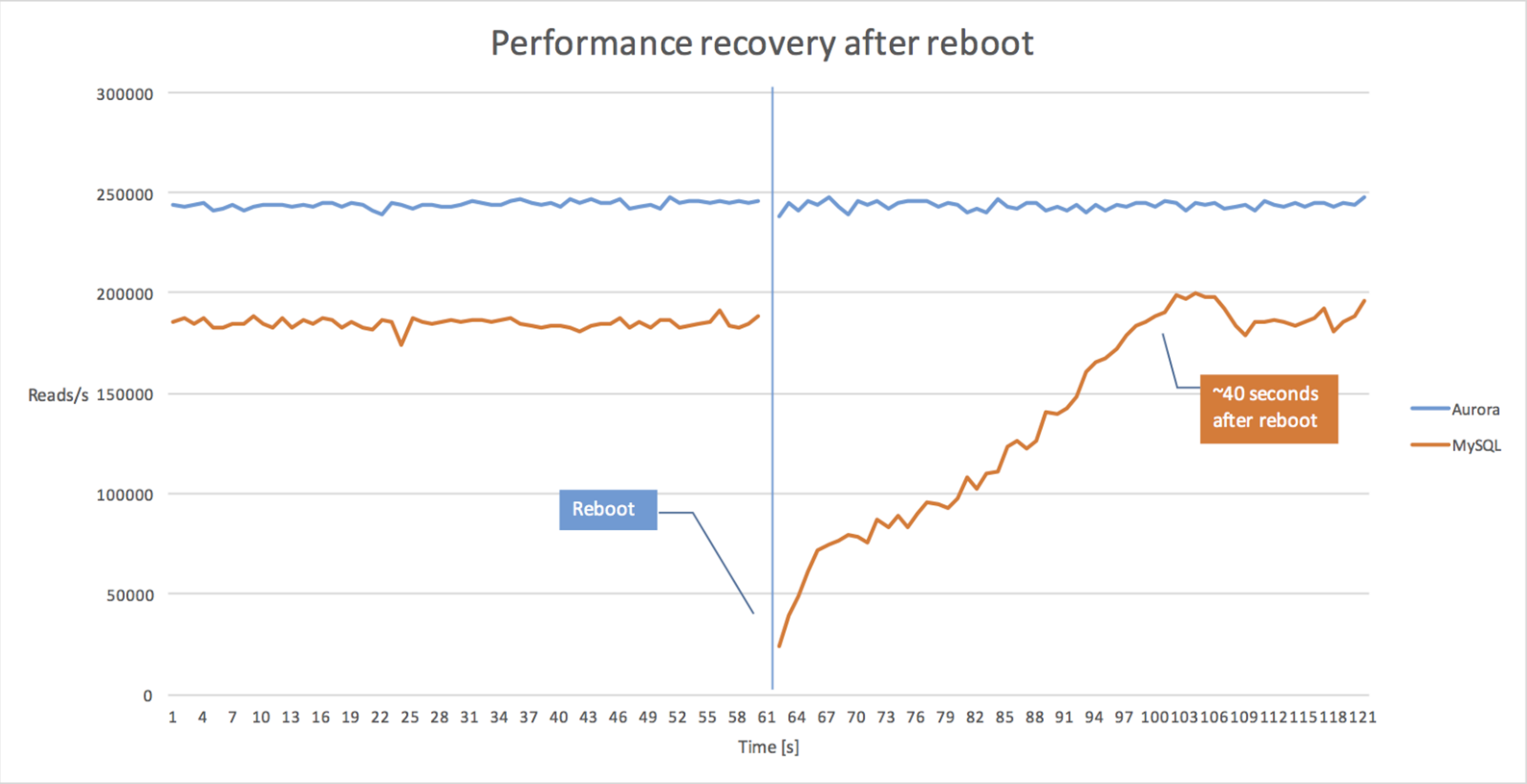
Read and write availability

多达15个可提升为主节点的读副本



- 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; customer specifiable failover order

重启后恢复到优化性能状态



Crash 恢复时间

Setup	Recovery Time
Aurora	00:00:20
Percona Server 5.7	00:02:30
Percona Server 5.7 (performance)	00:54:00
Percona Server 5.7 (performance, cold EBS)	>24 hours

recovery comes into play in all cases such as:

- **Reboots,**
- **Failovers,**
- **Point in time restores,**
- **Snapshot restores,**
- **Creating new replicas.**

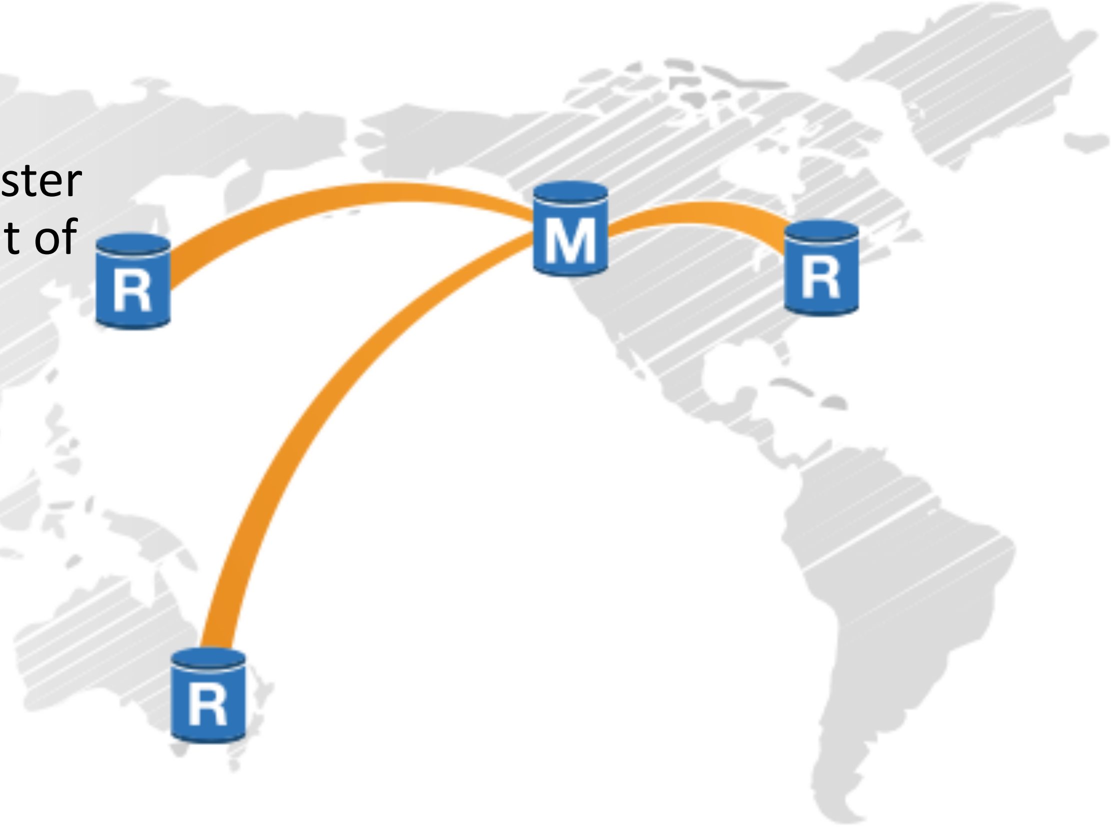
跨区域的读副本

更快的灾难恢复并增强数据本地化访问能力

- Promote read-replica to a master for faster recovery in the event of disaster

- Bring data close to your customer’s applications in different regions

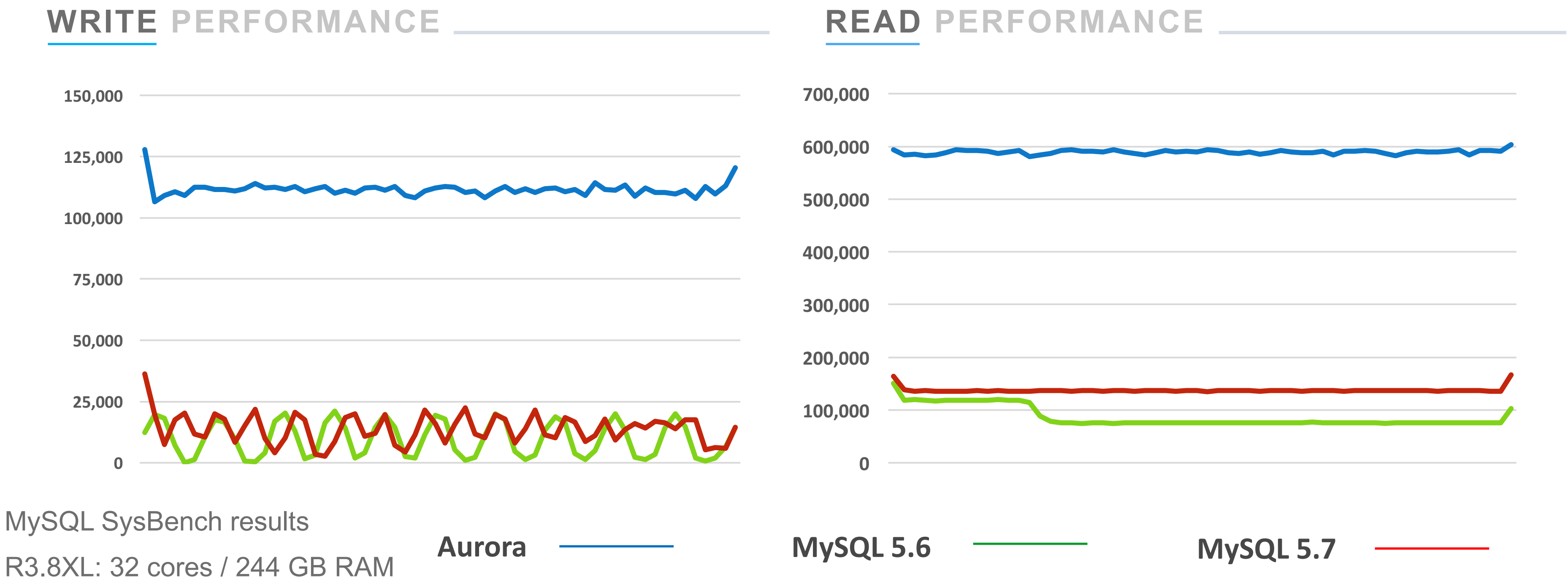
- Promote to a master for easy migration



Amazon Aurora 更快…

比 MySQL 快5倍以上

比RDS MySQL 5.6 & 5.7 快5倍



MySQL SysBench results

R3.8XL: 32 cores / 244 GB RAM

Five times higher throughput than stock MySQL based on industry standard benchmarks.

重现性能测试结果的步骤

1 Create an Amazon VPC (or use an existing one).

2 Create four EC2 R3.8XL client instances to run the SysBench client. All four should be in the same AZ.

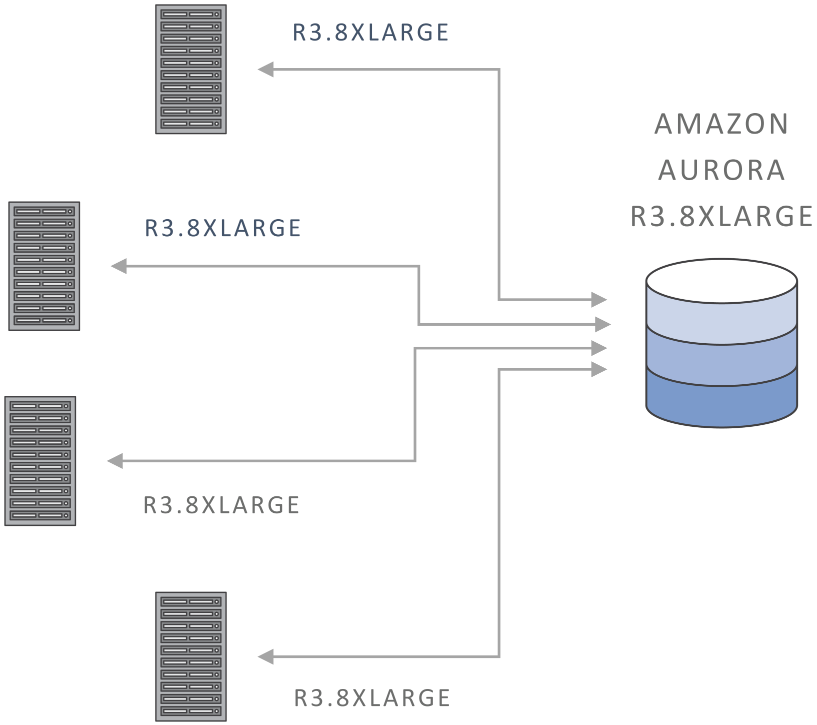
3 Enable enhanced networking on your clients.

4 Tune your Linux settings (see whitepaper).

5 Install Sysbench version 0.5.

6 Launch a r3.8xlarge Amazon Aurora DB instance in the same VPC and AZ as your clients.

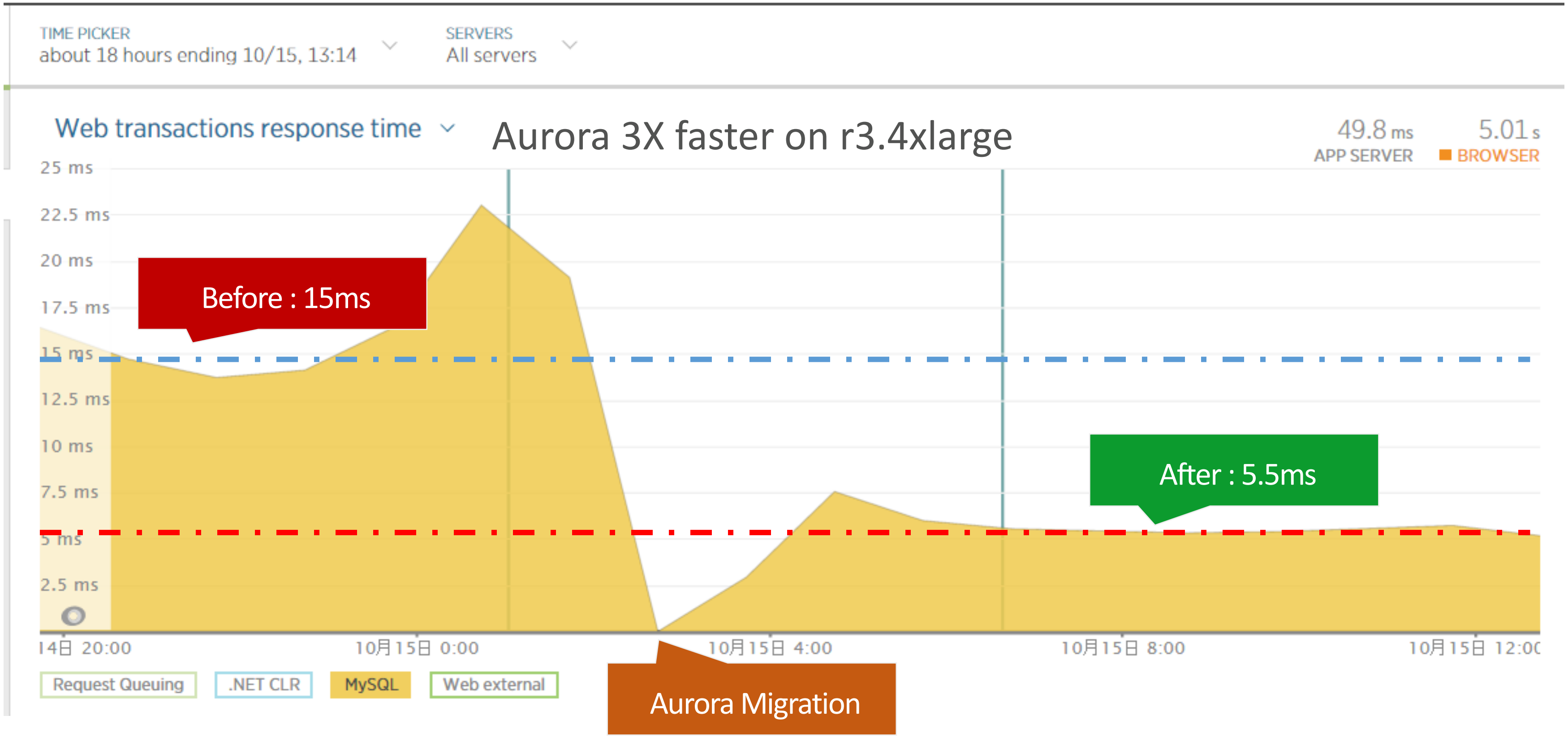
7 Start your benchmark!



https://d0.awsstatic.com/product-marketing/Aurora/RDS_Aurora_Performance_Assessment_Benchmarking_v1-2.pdf

真实环境数据 – gaming workload

Aurora vs. RDS MySQL – r3.4XL



TIME PICKER
about 18 hours ending 10/15, 13:14

SERVERS
All servers

Aurora 3X faster on r3.4xlarge

Before : 15ms

After : 5.5ms

Aurora Migration

Amazon Aurora 如何实现高性能

How Does Amazon Aurora Achieve High Performance?

DO LESS WORK

Do fewer IOs

Minimize network packets

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

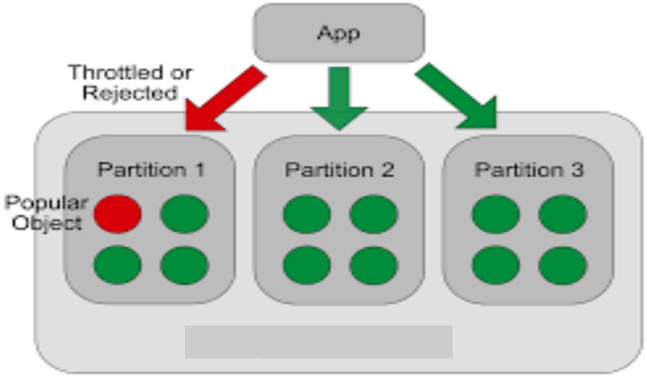
HIGH-THROUGHPUT PROCESSING NEEDS CPU AND MEMORY OPTIMIZATIONS

使用场景：大规模并发事件存储

For messaging, gaming, IoT

New Aurora-backed data store reduces operational costs by **40%**

- The cost of reading data (**70%** of user traffic) almost eliminated due to memory-bound nature of the workload.
- Only pay for IO used, not provisioned. Also, Aurora does automatic hot spot management. So, no need to over provision IOPS based on IO requirements of hottest partition.



Customer, a global mobile messaging platform, was using NoSQL key-value database for user messages:

- **~22 million** accesses per hour (70% read, 30% write) - billing grows linearly with the traffic.
- Scalability bottleneck where certain portions (partitions) of data became “hot” and overloaded with requests.

Amazon Aurora 易于使用

自动的存储管理, 安全与合规支持, 高级监控功能, 数据库迁移.

使用 SQL 语句模拟故障

- To cause the failure of a component at the database node:

```
ALTER SYSTEM CRASH [{INSTANCE | DISPATCHER | NODE}]
```

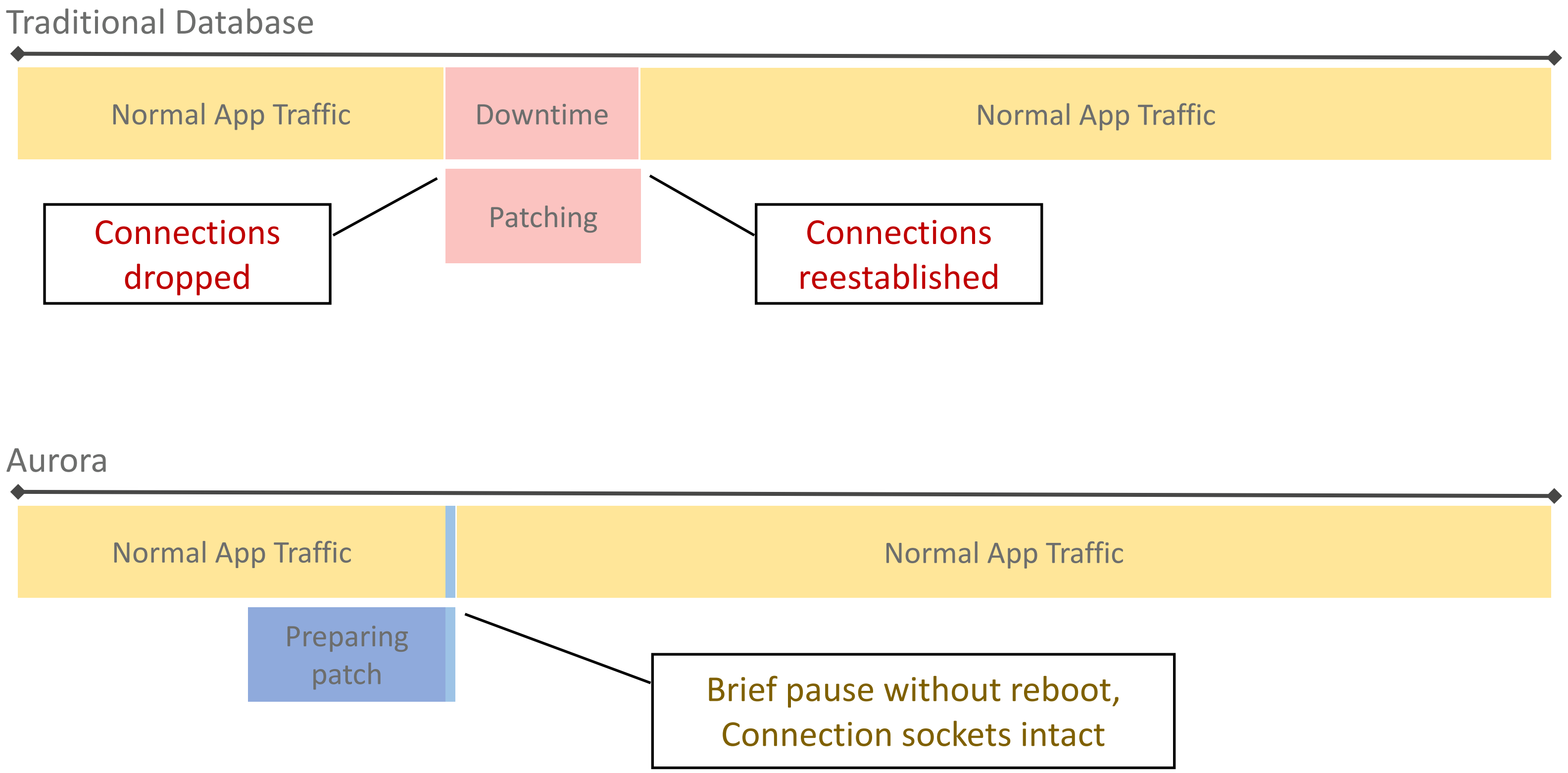
- To simulate the failure of disks:

```
ALTER SYSTEM SIMULATE percent_failure DISK failure_type IN
[DISK index | NODE index] FOR INTERVAL interval
```

- To simulate the failure of networking:

```
ALTER SYSTEM SIMULATE percent_failure NETWORK failure_type
[TO {ALL | read_replica | availability_zone}] FOR INTERVAL interval
```

零停机安装补丁包



Traditional Database

Normal App Traffic

Downtime

Normal App Traffic

Connections dropped

Patching

Connections reestablished

Aurora

Normal App Traffic

Normal App Traffic

Preparing patch

Brief pause without reboot, Connection sockets intact

在线 DDL 性能

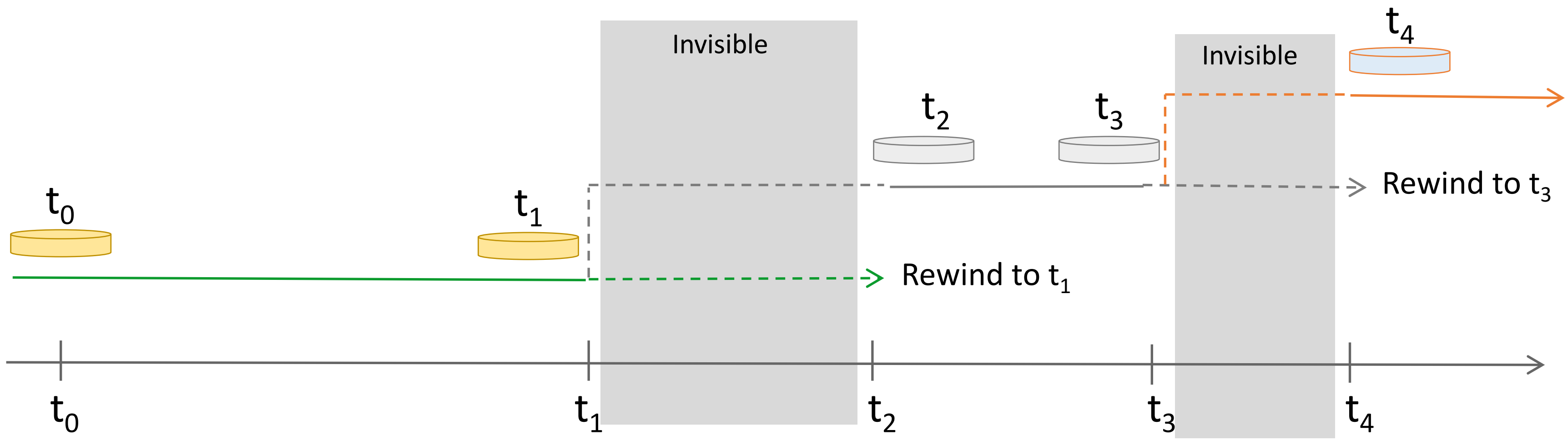
On r3.large

	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

在线指定时间点恢复



- Online point-in-time restore is a quick way to bring the database to a particular point in time without having to restore from backups
  - Rewinding the database to quickly recover from unintentional DML/DDL operations.
  - Rewind multiple times to determine the desired point-in-time in the database state. For example, quickly iterate over schema changes without having to restore multiple times.

THANKS

SequeMedia 盛拓传媒 IT168.com ITPUB ChinaUnix.net