Databases on AWS

Chetan Agrawal, Solutions Architect

Deven Suri, Account Manager

Agenda

- AWS Database Services
- Traditional vs AWS Data services model
- Amazon RDS
- Amazon Aurora
- Amazon DynamoDB
- Amazon ElastiCache
- Amazon Neptune
- Amazon DocumentDB
- Amazon QLDB
- Amazon Timestream

Common data categories and use cases

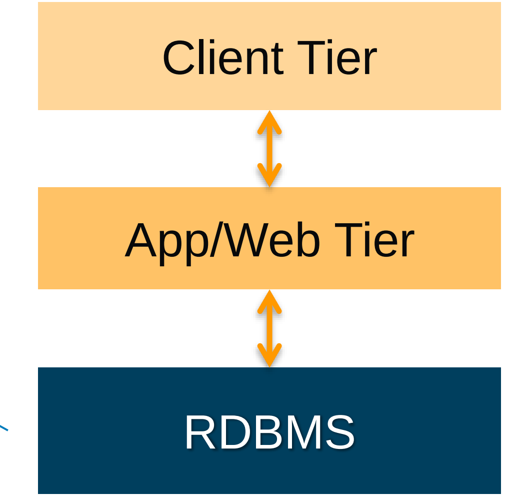
Relational	Key-value	Document	In-memory	Graph	Time-series	Ledger	Warehouse
Referential integrity, ACID transactions, schema-on-write	High throughput, low-latency reads and writes, endless scale	Store JSON documents with quick access, query on any attribute	Query by key with microsecond latency	Quickly and easily create and navigate relationships between data	Collect, store, and process data sequenced by time	Complete, immutable, and verifiable history of all changes to application data	High performance querying on large volumes of data
Lift and shift, ERP, CRM, finance	Real-time bidding, shopping cart, social, product catalog, customer preferences	Content management, personalization, mobile	Leaderboards, real-time analytics, caching	Fraud detection, social networking, recommendation engine	IoT applications, event tracking	Systems of record, supply chain, health care, registrations, financial	Analytics, Data Marts

Common data categories and use cases

Relational	Key-value	Document	In-memory	Graph	Time-series	Ledger	Warehouse
Referential integrity, ACID transactions, schema-on-write	High throughput, low-latency reads and writes, endless scale	Store JSON documents with quick access, query on any attribute	Query by key with microsecond latency	Quickly and easily create and navigate relationships between data	Collect, store, and process data sequenced by time	Complete, immutable, and verifiable history of all changes to application data	High performance querying on large volumes of data
RDS	DynamoDB	DocumentDB	ElastiCache Redis Memcached	Neptune	Timestream Preview	QLDB	Redshift

Traditional Database Architecture

one database for all workloads



Client Tier

App/Web Tier

RDBMS

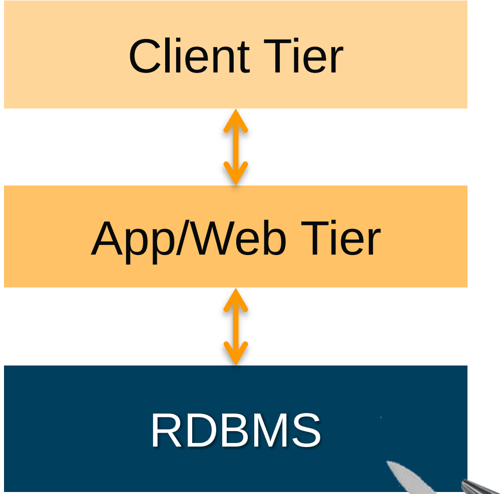
Traditional Database Architecture

Key-value access

Complex queries

OLAP transactions

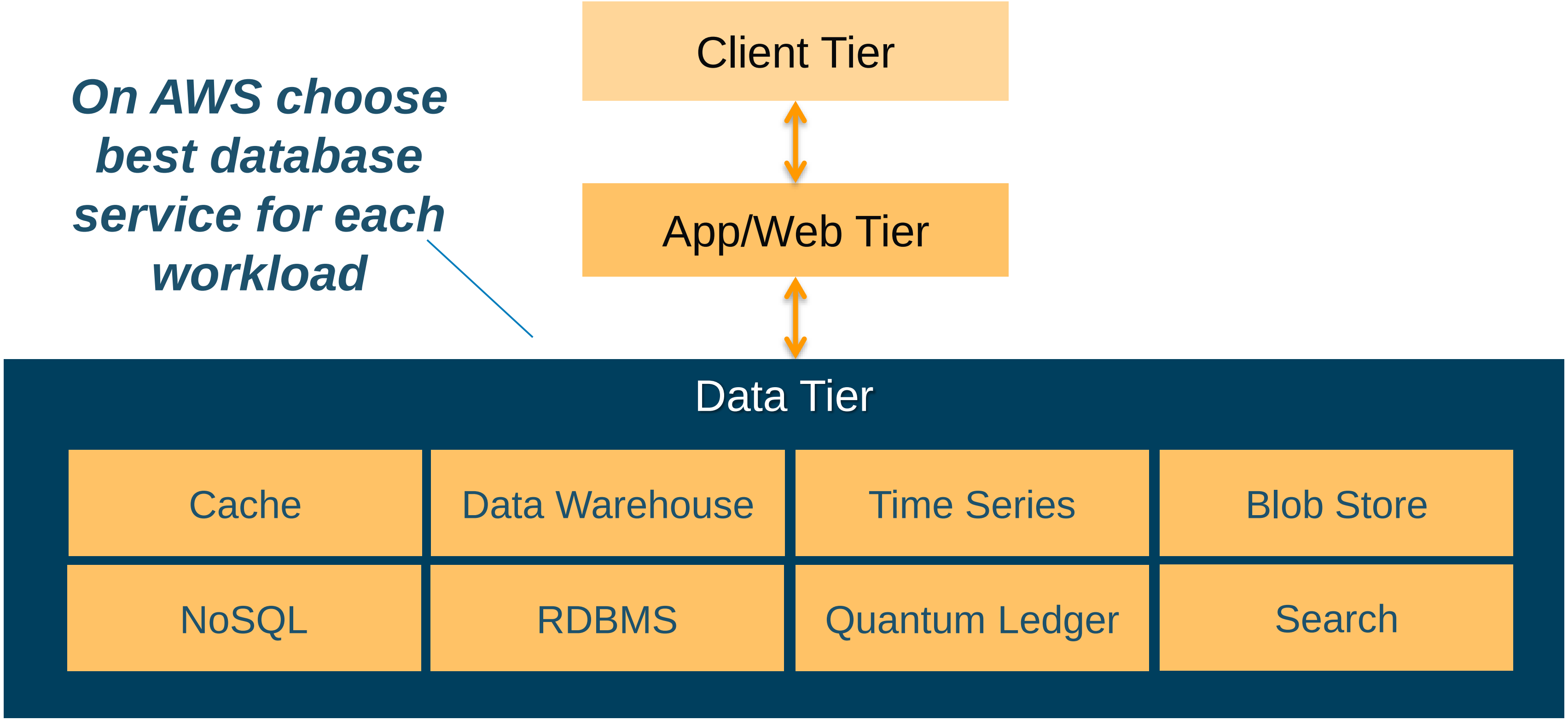
Analytics



All forced into the relational database

AWS Data Tier Architecture

On AWS choose best database service for each workload



Client Tier

App/Web Tier

Data Tier

Cache

Data Warehouse

Time Series

Blob Store

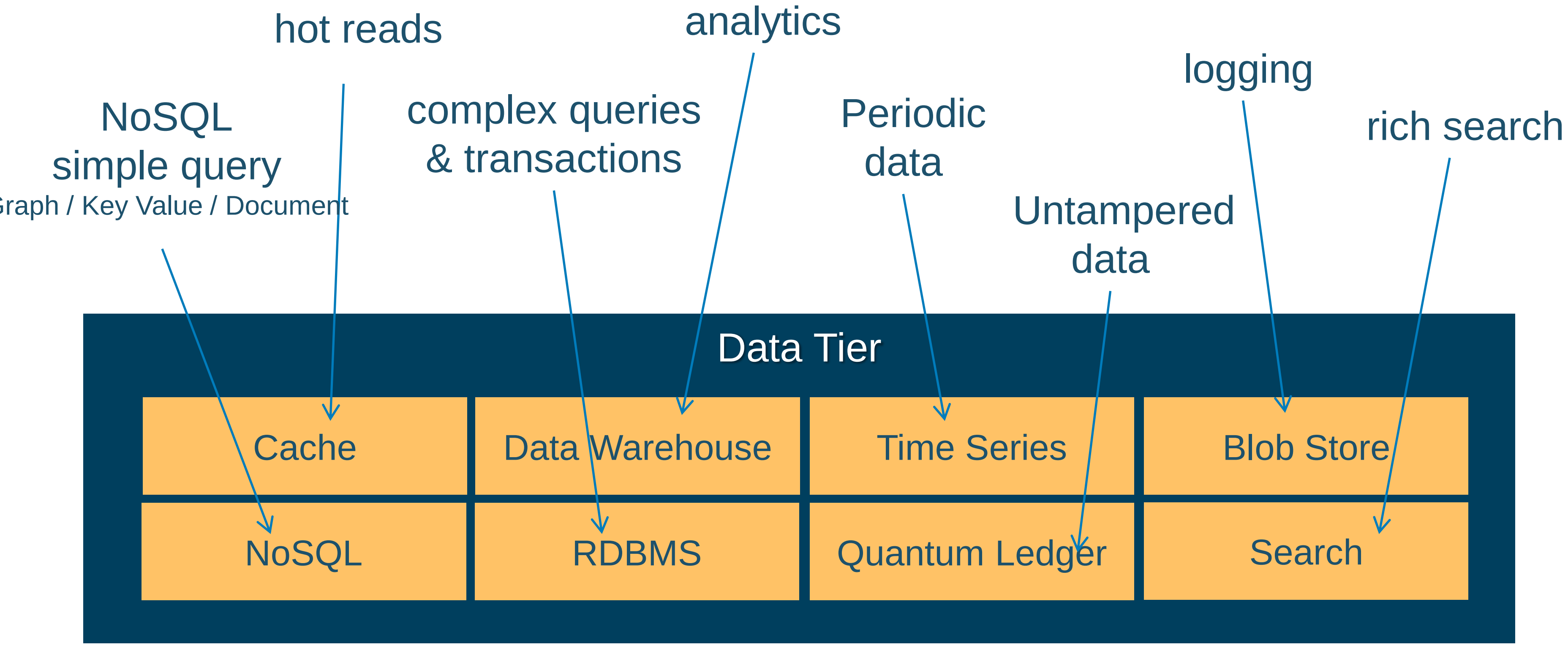
NoSQL

RDBMS

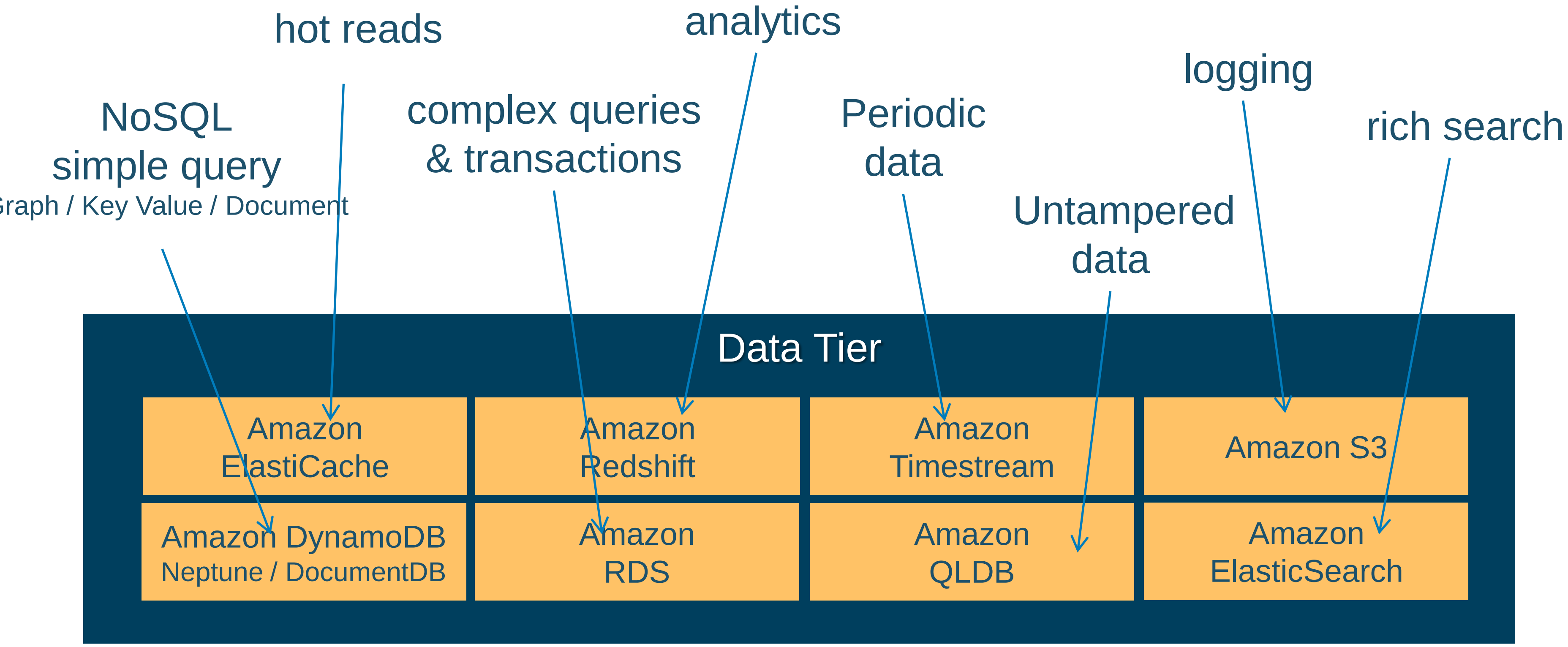
Quantum Ledger

Search

Workload Driven Data Store Selection



AWS Database Services for the Data Tier



NoSQL
simple query
Graph / Key Value / Document

hot reads

complex queries
& transactions

analytics

Periodic
data

Untampered
data

logging

rich search

Data Tier

Amazon
ElastiCache

Amazon
Redshift

Amazon
Timestream

Amazon S3

Amazon DynamoDB
Neptune / DocumentDB

Amazon
RDS

Amazon
QLDB

Amazon
ElasticSearch

Amazon RDS

Managed relational database service with a choice of popular database engines

Amazon Aurora · MySQL · PostgreSQL · MariaDB · Microsoft SQL Server · ORACLE

Easy to administer

Easily deploy and maintain hardware, OS and DB software; built-in monitoring

Performant & scalable

Scale compute and storage with a few clicks; minimal downtime for your application

Available & durable

Automatic Multi-AZ data replication; automated backup, snapshots, and failover

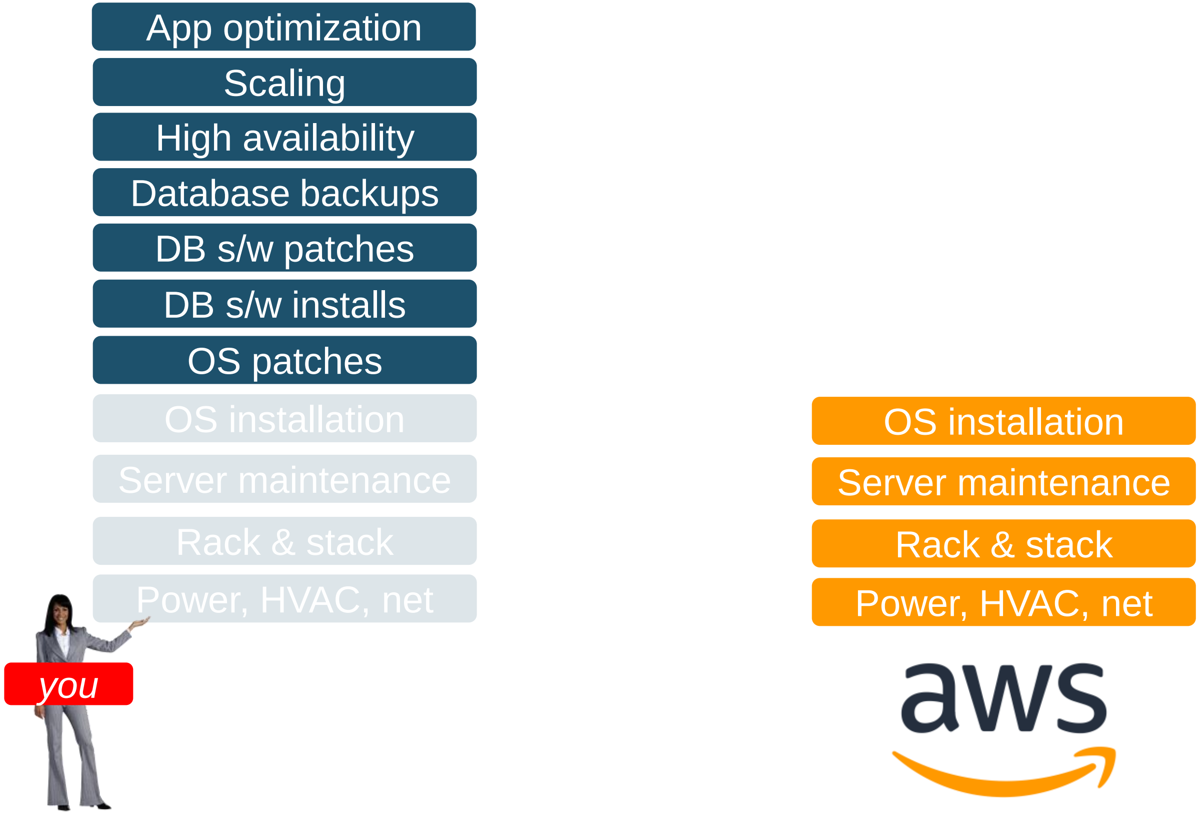
Secure and compliant

Data encryption at rest and in transit; industry compliance and assurance programs

If you host your databases on-premises…

- App optimization
- Scaling
- High availability
- Database backups
- DB s/w patches
- DB s/w installs
- OS patches
- OS installation
- Server maintenance
- Rack & stack
- Power, HVAC, net

If you host your databases in Amazon EC2…



App optimization

Scaling

High availability

Database backups

DB s/w patches

DB s/w installs

OS patches

OS installation

Server maintenance

Rack & stack

Power, HVAC, net

OS installation

Server maintenance

Rack & stack

Power, HVAC, net

If you choose Amazon RDS…

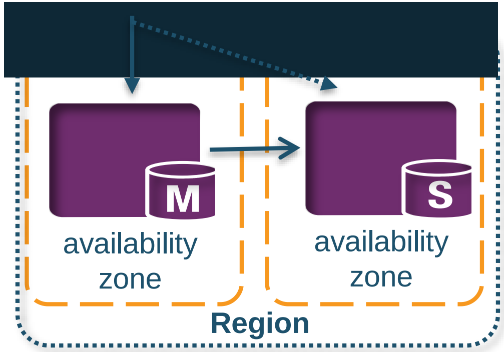
- App optimization
- Scaling
- High availability
- Database backups
- DB s/w patches
- DB s/w installs
- OS patches
- OS installation
- Server maintenance
- Rack & stack
- Power, HVAC, net

- Scaling
- High availability
- Database backups
- DB s/w patches
- DB s/w installs
- OS patches
- OS installation
- Server maintenance
- Rack & stack
- Power, HVAC, net

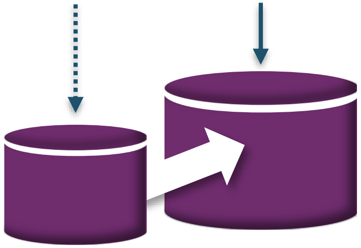
Key Amazon RDS Features

Amazon RDS Configuration	Improve Availability	Increase Throughput	Reduce Latency
Push-Button Scaling		✓	
Multi AZ	✓		
Read Replicas		✓	
Provisioned IOPS		✓	✓

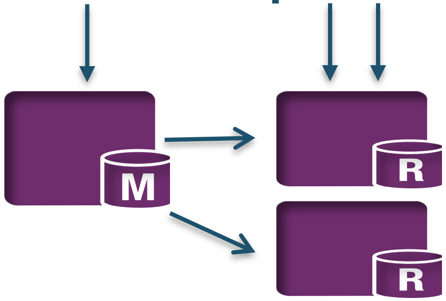
Multi-AZ



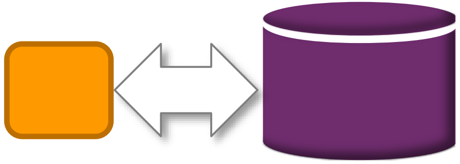
Push-Button Scaling



Read Replicas

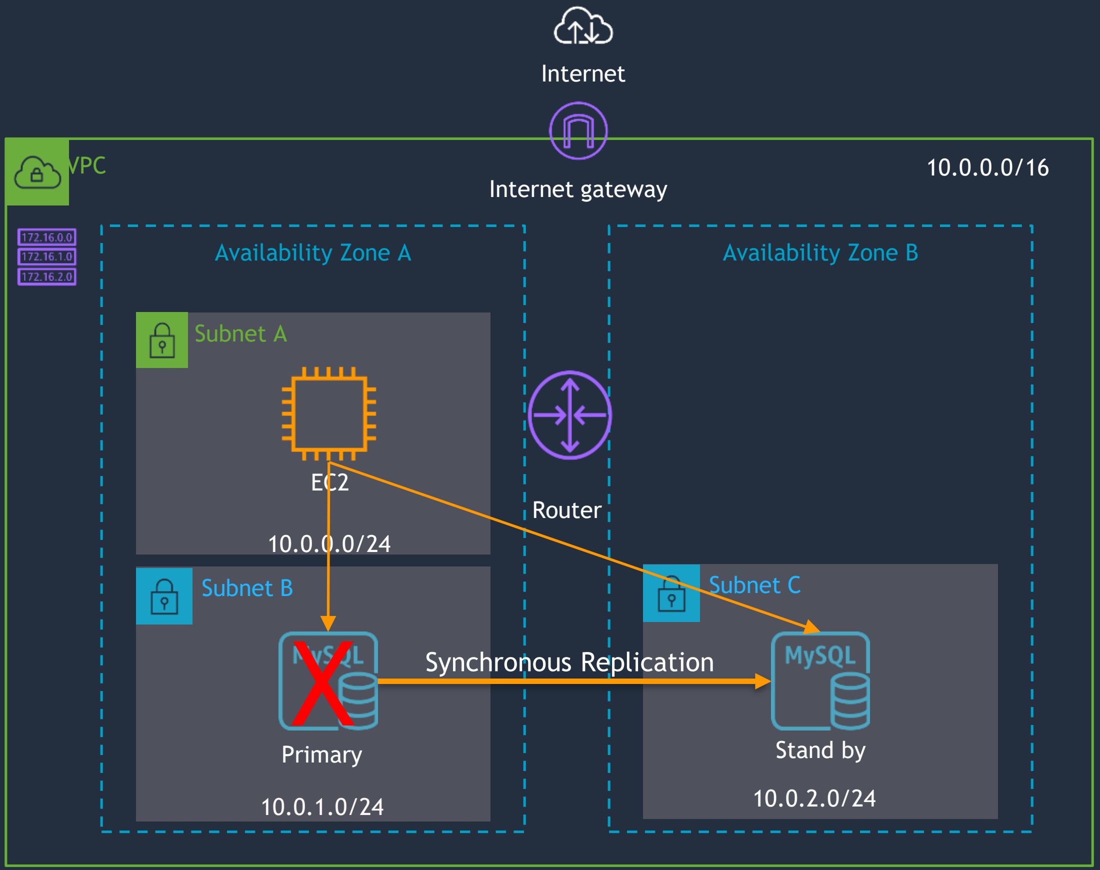


Provisioned IOPS



Demo – Multi-AZ RDS

- Launch MySQL DB in multiple Azs
- Connect to Primary
- Simulator the failure
- Check if secondary DB takes over automatically



Demo steps

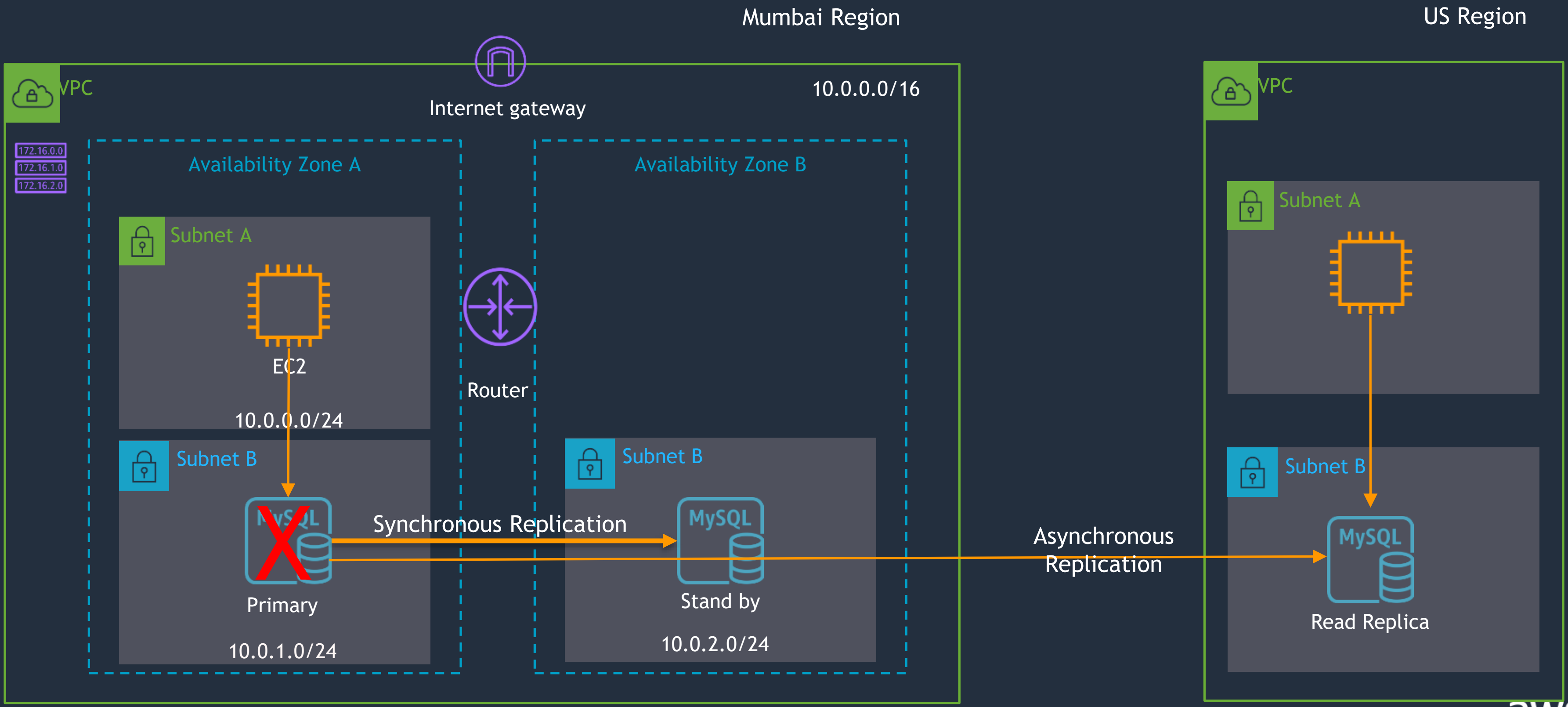
1. Connect to DB
2. Create table and insert some data

```
> create database test
> use test;
> create table amazon (name VARCHAR(30), id INT(2), phone VARCHAR(10), email VARCHAR(30));
> insert into amazon  values ('Chetan Agrawal', 3, '9x2x5x3x6x', 'agrcheta@amazon.com');
```

3. Simulate the failure in master (Reboot with failover)
4. Wait and check if failover happens automatically

```
$while true; do host database-1.xxxxxx.ap-south-1.rds.amazonaws.com; sleep 3; done
```

Exercise – Read Replica across Region



Amazon Aurora

MySQL and PostgreSQL compatible relational database built for the cloud

Performance and availability of commercial-grade databases at 1/10th the cost

Performance & scalability

5x throughput of standard MySQL and 3x of standard PostgreSQL; scale-out up to15 read replicas

Availability & durability

Fault-tolerant, self-healing storage; six copies of data across three AZs; continuous backup to S3

Highly secure

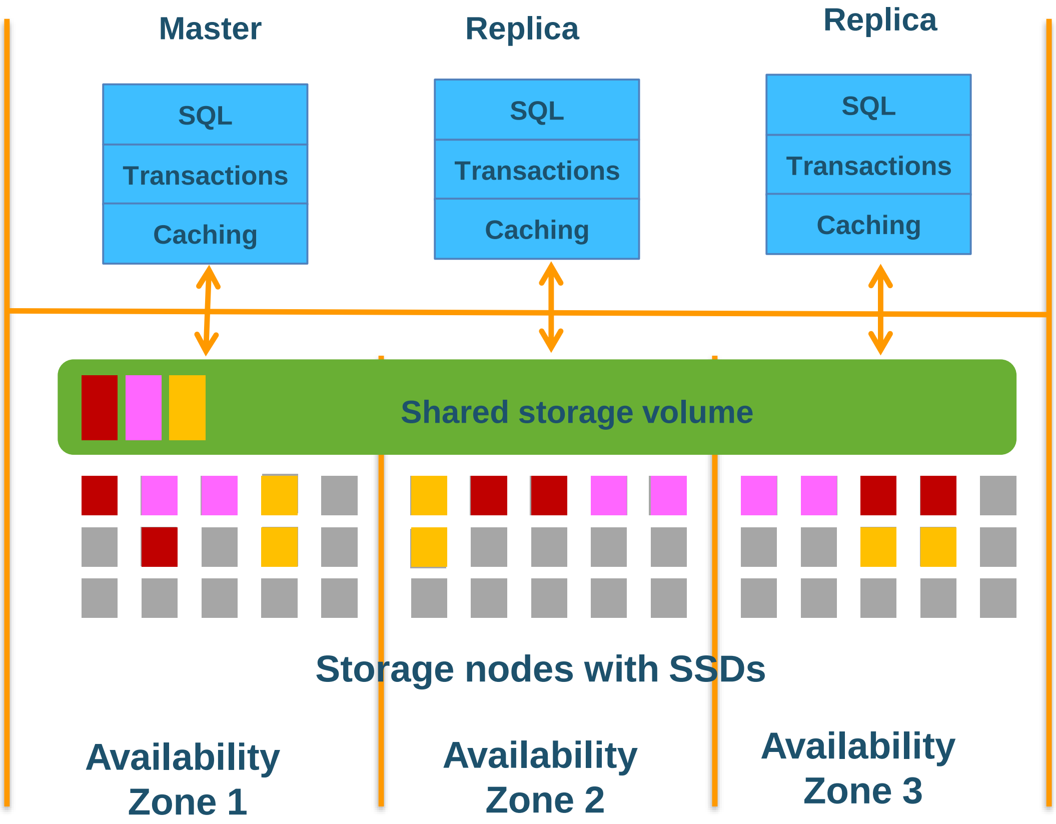
Network isolation, encryption at rest/transit

Fully managed

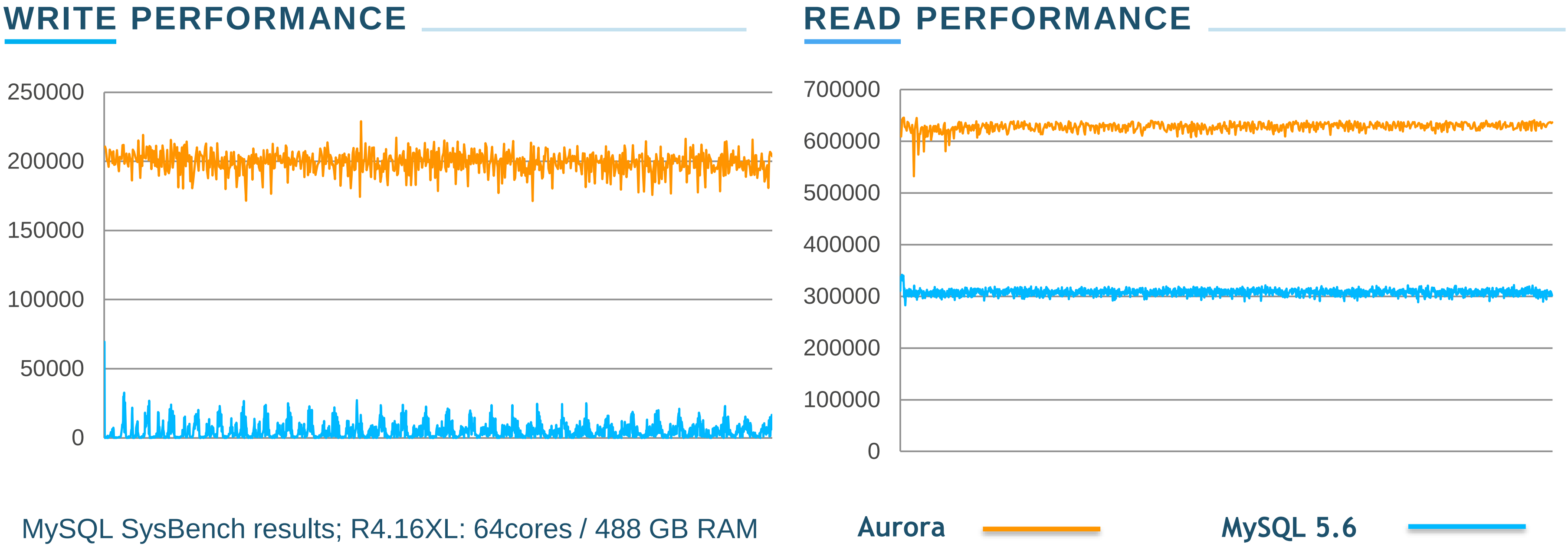
Managed by RDS: no hardware provisioning, software patching, setup, configuration, or backups

Scale-out, distributed, multi-tenant 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
- Master and replicas all point to the same storage



Aurora MySQL performance

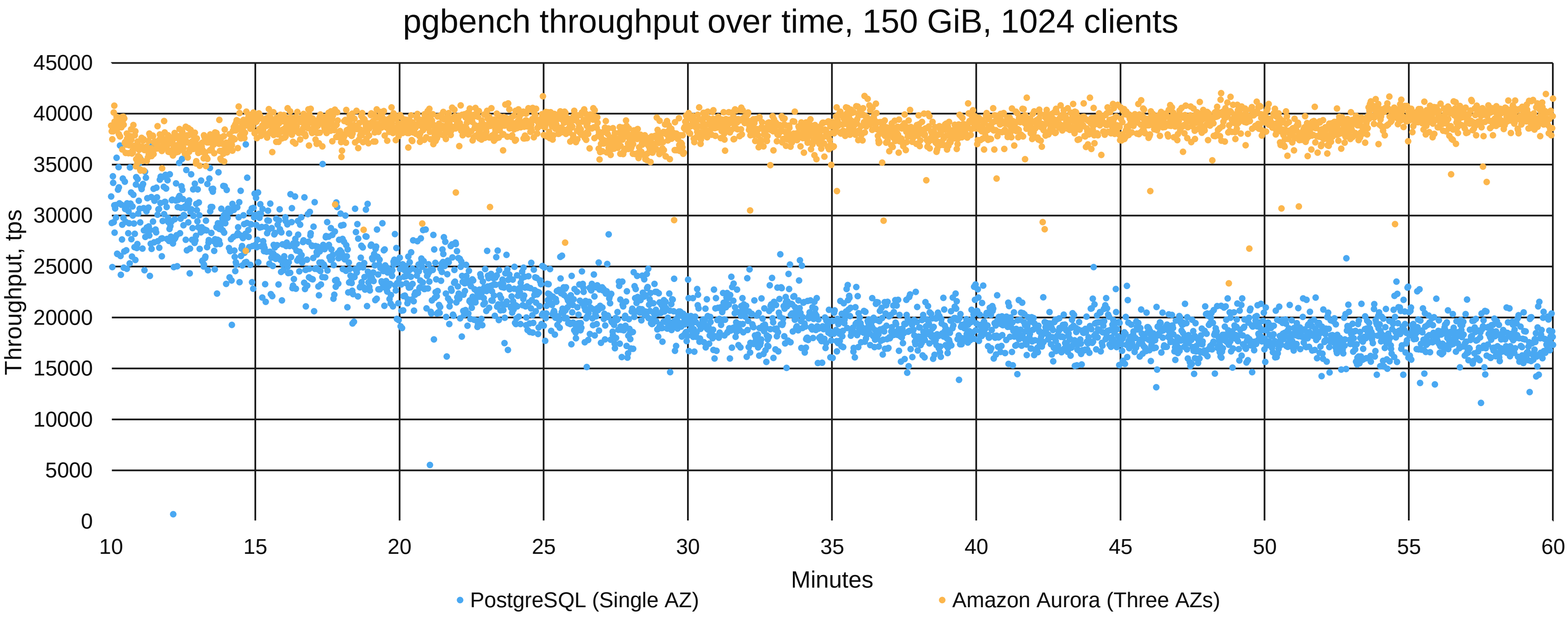


MySQL SysBench results; R4.16XL: 64cores / 488 GB RAM

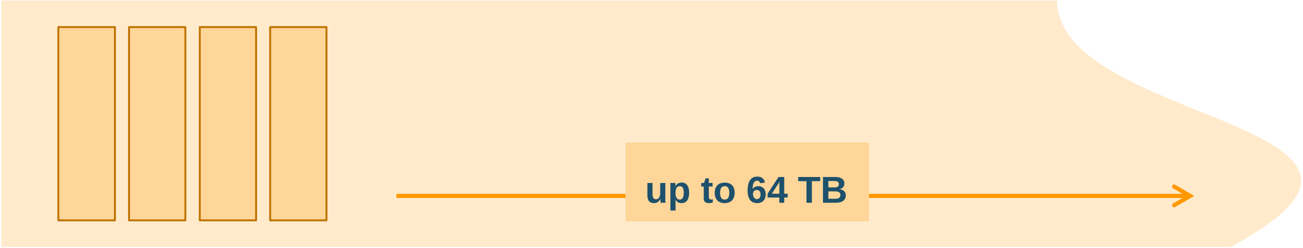
Aurora read write throughput compared to MySQL 5.6 based on industry standard benchmarks.

Aurora PostgreSQL performance

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



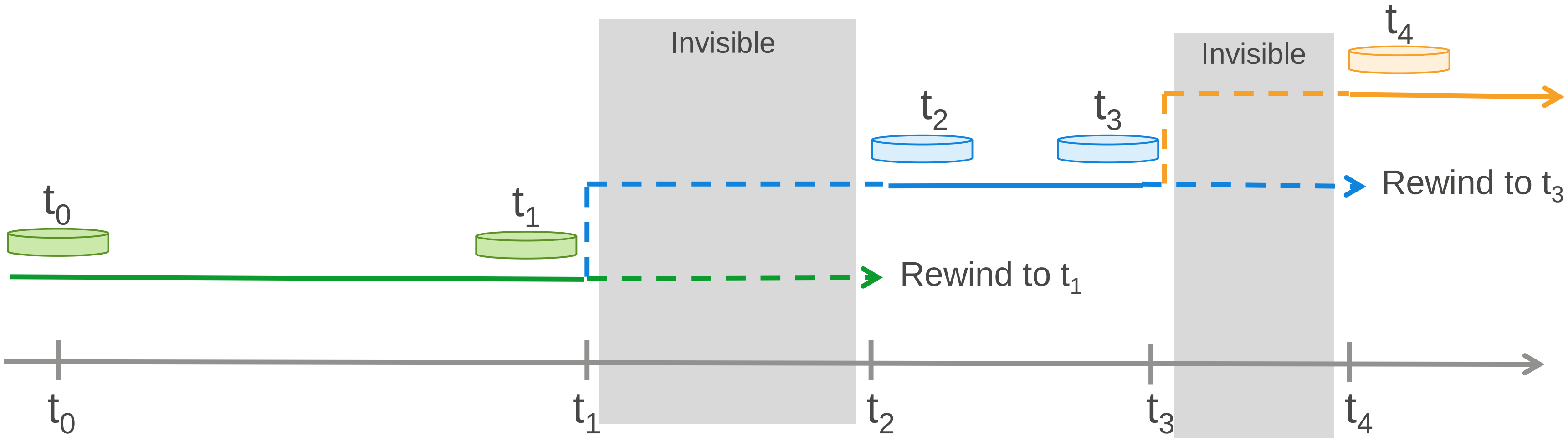
…and more



Up to 64TB of storage – auto-incremented in 10GB units

- Automatic storage scaling up to 64 TB—no performance impact
- Continuous, incremental backups to Amazon S3
- Instantly create user snapshots—no performance impact
- Automatic restriping, mirror repair, hot spot management, encryption

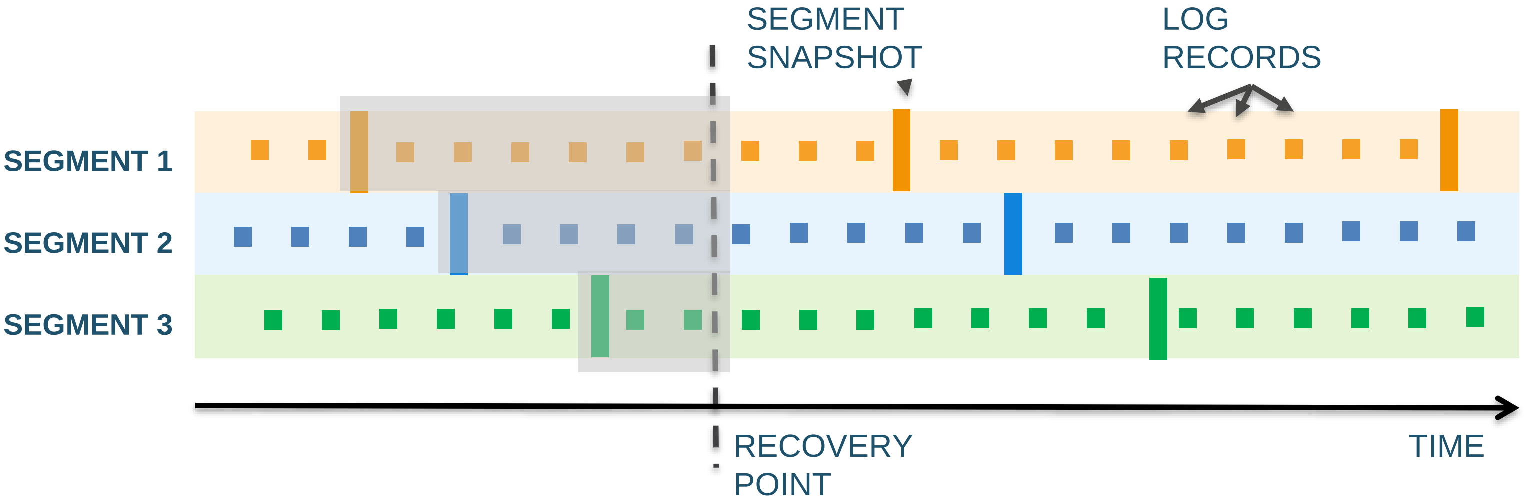
Database backtrack



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

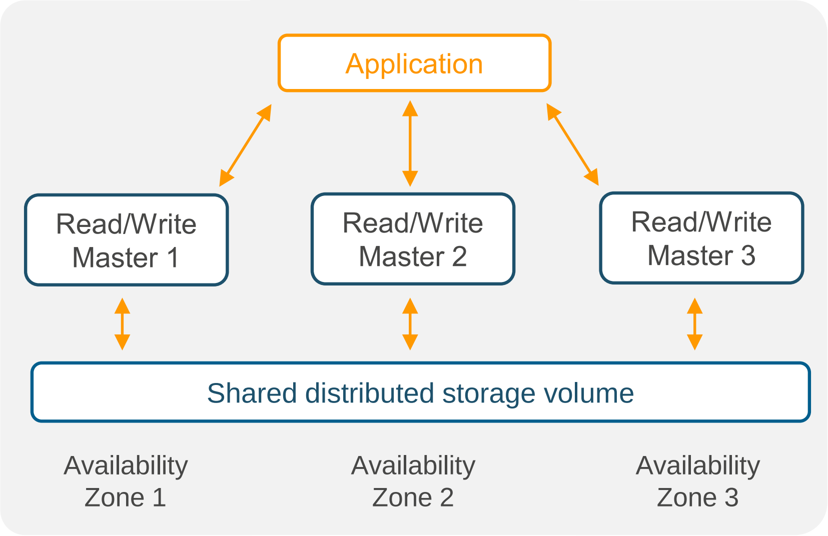
For backtrack, we identify the appropriate segment snapshots

Apply log streams to segment snapshots in parallel and asynchronously

Aurora Multi-Master

First relational database service with scale-out reads and writes across multiple data centers

Scale out both reads **and writes**



Zero application downtime from ANY instance failure

Zero application downtime from ANY AZ failure

Faster write performance and higher scale

Global database

Faster disaster recovery and enhanced data locality

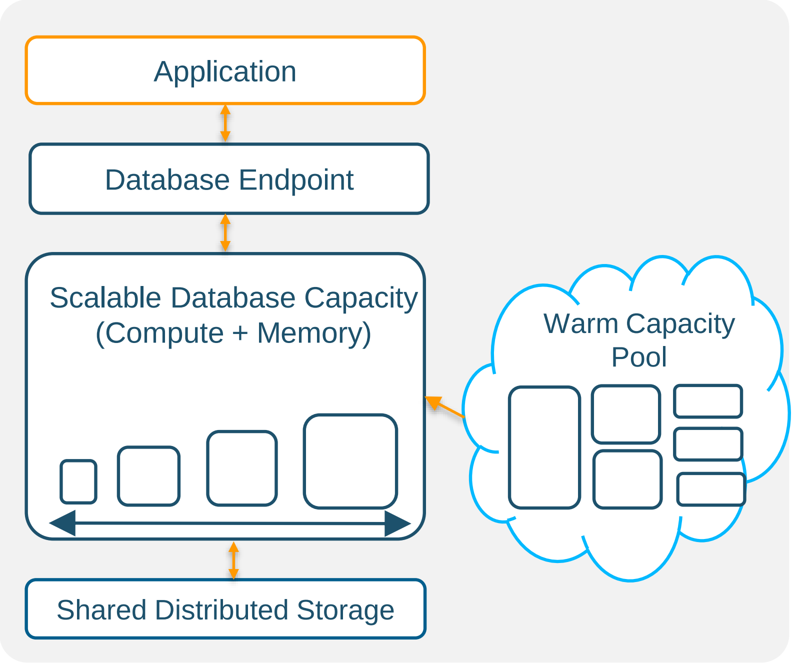
Promote read-replica to a master for faster recovery **in the event of disaster**

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

Promote to a master for **easy migration**

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

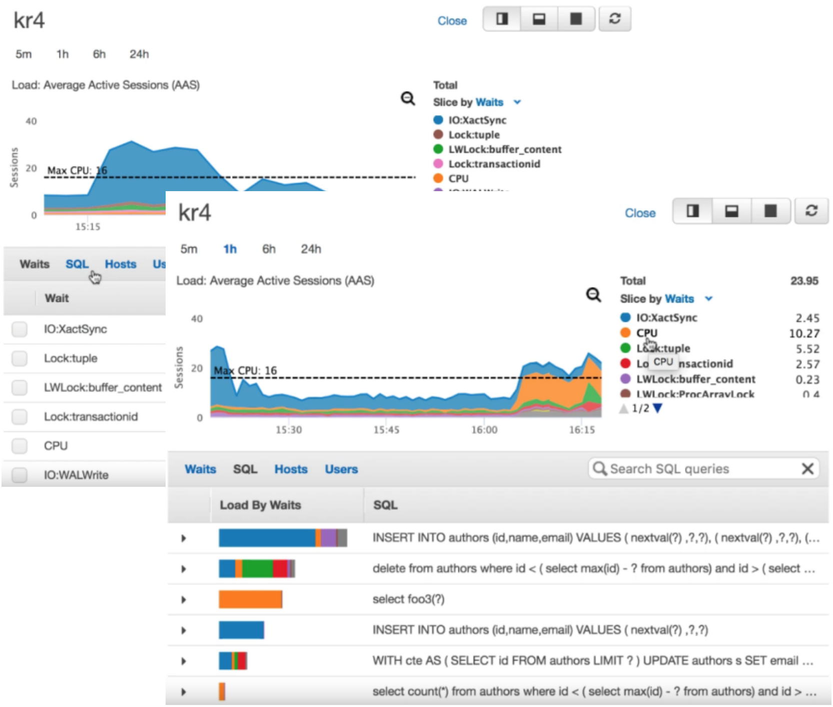
Automatically scales with no instances to manage

Pay per second for the database capacity you use

Performance Insights for Aurora

Analyze and troubleshoot your database performance

- Supports PostgreSQL and MySQL
- Expands on existing Amazon RDS monitoring features to analyze issues and performance
- Easy bottleneck identification – keep track of performance metrics such as high CPU consumption, lock waits, I/O latency, and SQL statements



	Wait
□	IO:XactSync
□	Lock:tuple
□	LWLock:buffer_content
□	Lock:transactionid
□	CPU
□	IO:WALWrite

Total	23.95
Slice by Waits	
● IO:XactSync	2.45
● CPU	10.27
● Lock:tuple	5.52
● Lock:transactionid	2.57
● LWLock:buffer_content	0.23
● LWLock:ProcArrayLock	0.4

Load By Waits	SQL
	INSERT INTO authors (id,name,email) VALUES (nextval(?) ,?,?), (nextval(?) ,?,?), (...
	delete from authors where id < (select max(id) - ? from authors) and id > (select ...
	select foo3(?)
	INSERT INTO authors (id,name,email) VALUES (nextval(?) ,?,?)
	WITH cte AS (SELECT id FROM authors LIMIT ?) UPDATE authors s SET email ...
	select count(*) from authors where id < (select max(id) - ? from authors) and id > ...

Amazon DynamoDB

NoSQL database

Seamless scalability

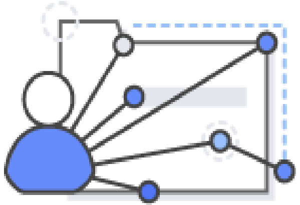
Zero admin

Single-digit millisecond latency

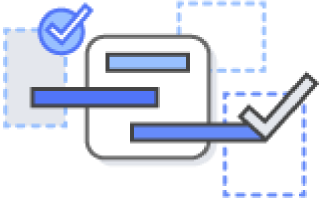
Multi-Master

Multi-Region

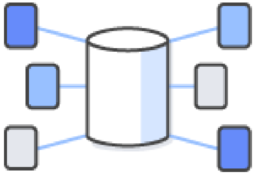
Amazon DynamoDB



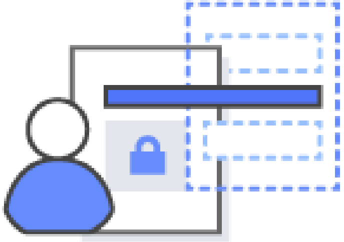
Fully managed



Consistently fast at any scale



Highly available
and durable



Secure

Integrates with AWS Lambda,
Amazon Redshift, and more

Cost-effective

Highly available and durable

Built for high durability

WRITES

3-way replication
Persisted to disk
(Custom SSD)

Designed to support
99.99%
of availability

READS

Strongly or eventually consistent
No latency trade-off

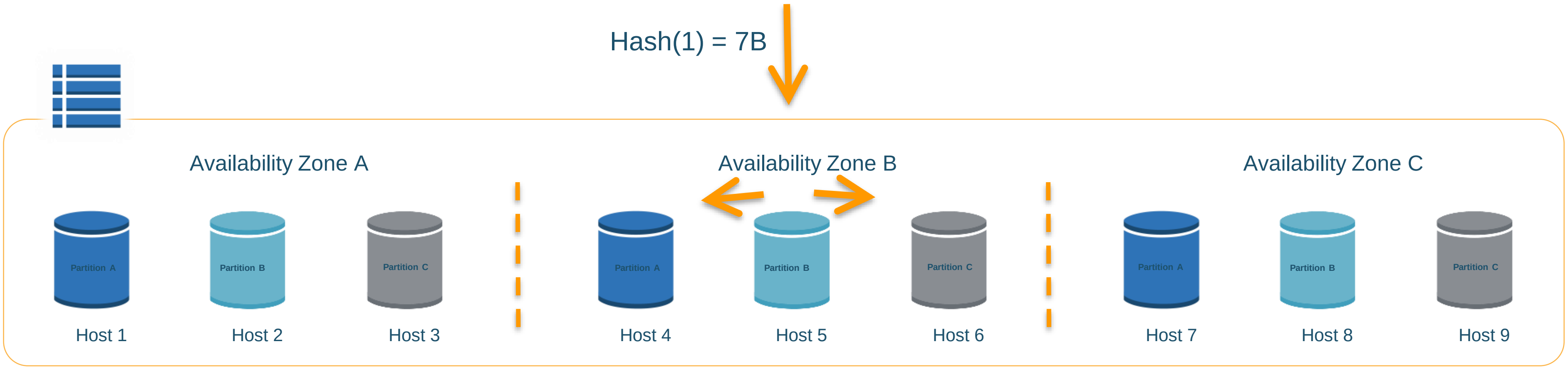
Data is always replicated to
three Availability Zones

Highly available and durable

OrderId: 1
CustomerId: 1
ASIN: [B00X4WHP5E]

3-way replication

Data is always replicated to three Availability Zones



CustomerOrdersTable

Backup and restore

The only cloud database to provide on-demand and continuous backups

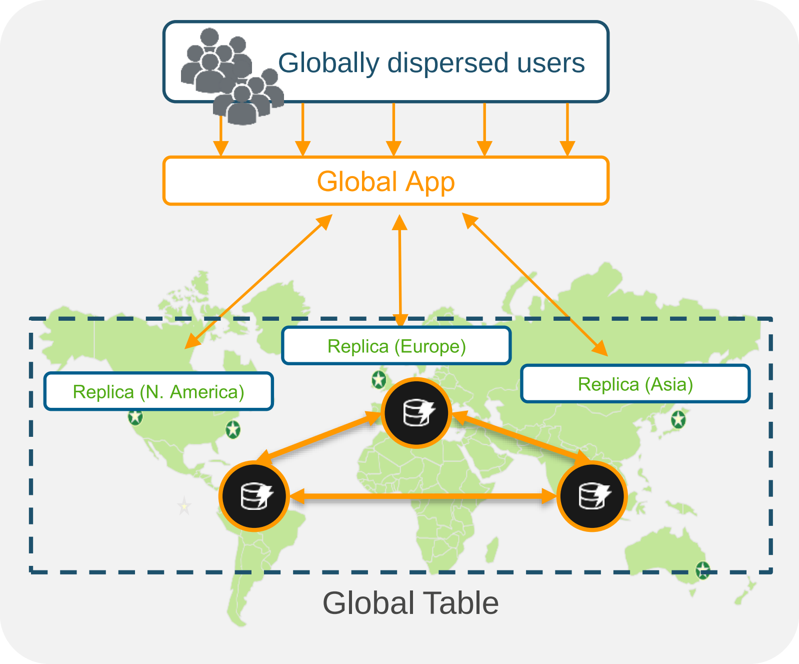
On-demand backups for long-term data archival and compliance

Point in time restore for short term retention and data corruption protection (35 days)

Point in time recovery with restore times in a few hours depending on table size

Global Tables

The first fully-managed, multi-master, multi-region database



Build high performance, globally distributed applications

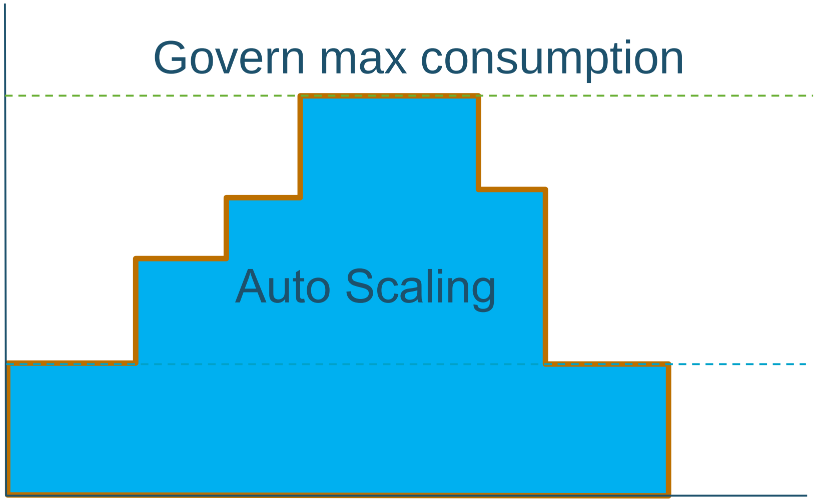
Low latency reads & writes to locally available tables

Disaster proof with multi-region redundancy

Easy to setup and no application re-writes required

Capacity managed for you

Provisioned

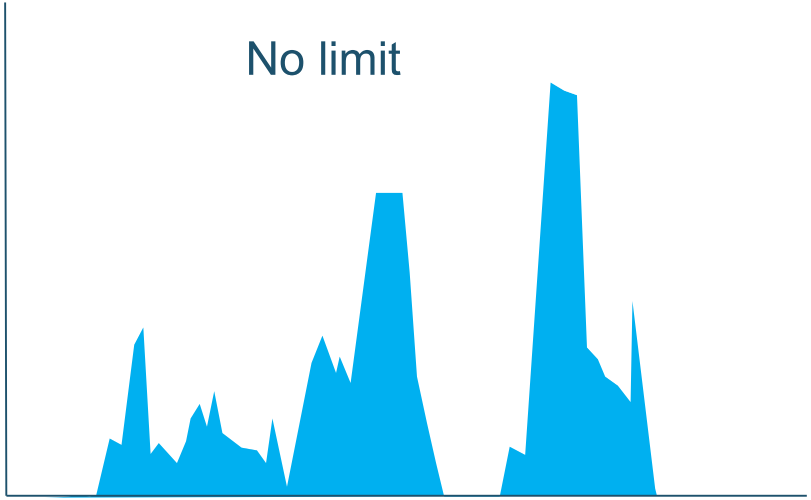


Govern max consumption

Auto Scaling

Set a minimum

On-Demand

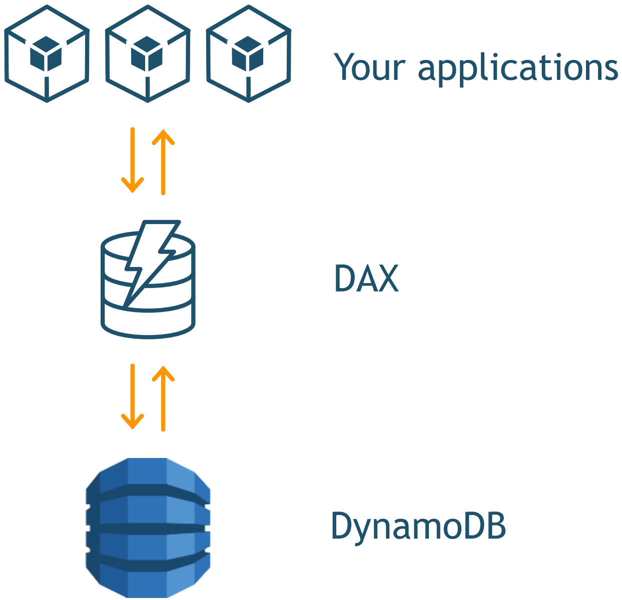


No limit

Start at zero

DynamoDB Accelerator (DAX)

High performance



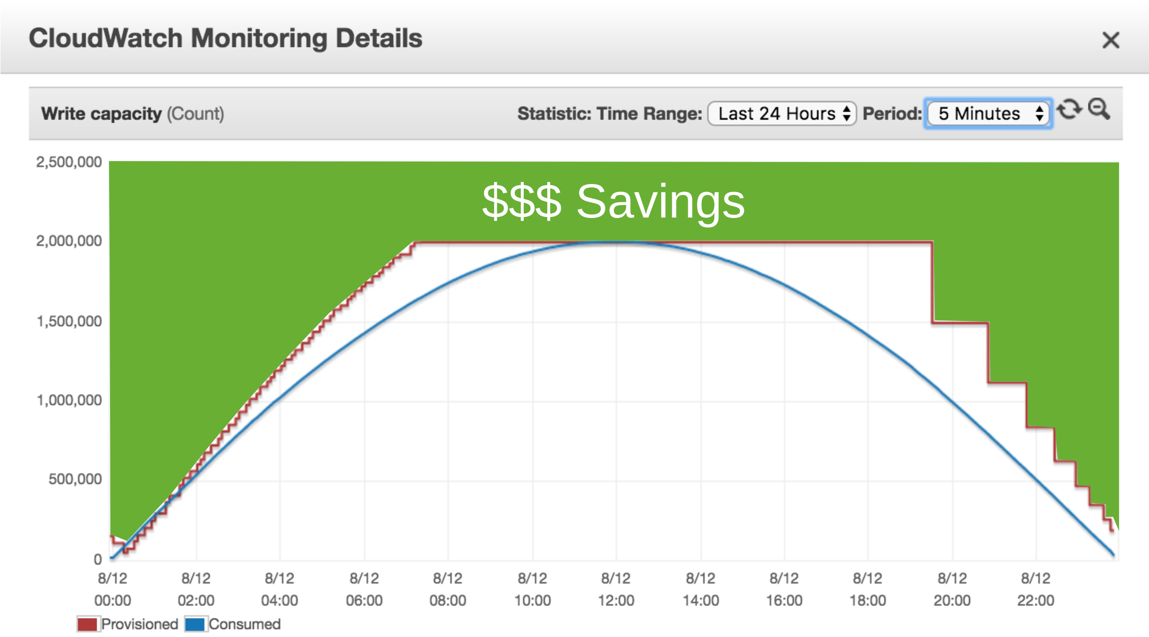
Fully managed, highly available cache for DynamoDB

Even faster—microsecond latency

Scales to millions of requests per second

API compatible

Fully managed auto scaling



CloudWatch Monitoring Details

Write capacity (Count)

Statistic: Time Range: Last 24 Hours Period: 5 Minutes

$$$ Savings

Automated scaling policies

Scales up when you need it

Scales down when you don’t

Scheduled auto scaling

NoSQL vs. SQL for a new app: how to choose?

Want simplest possible DB management?

Want app to manage DB integrity?

Amazon DynamoDB

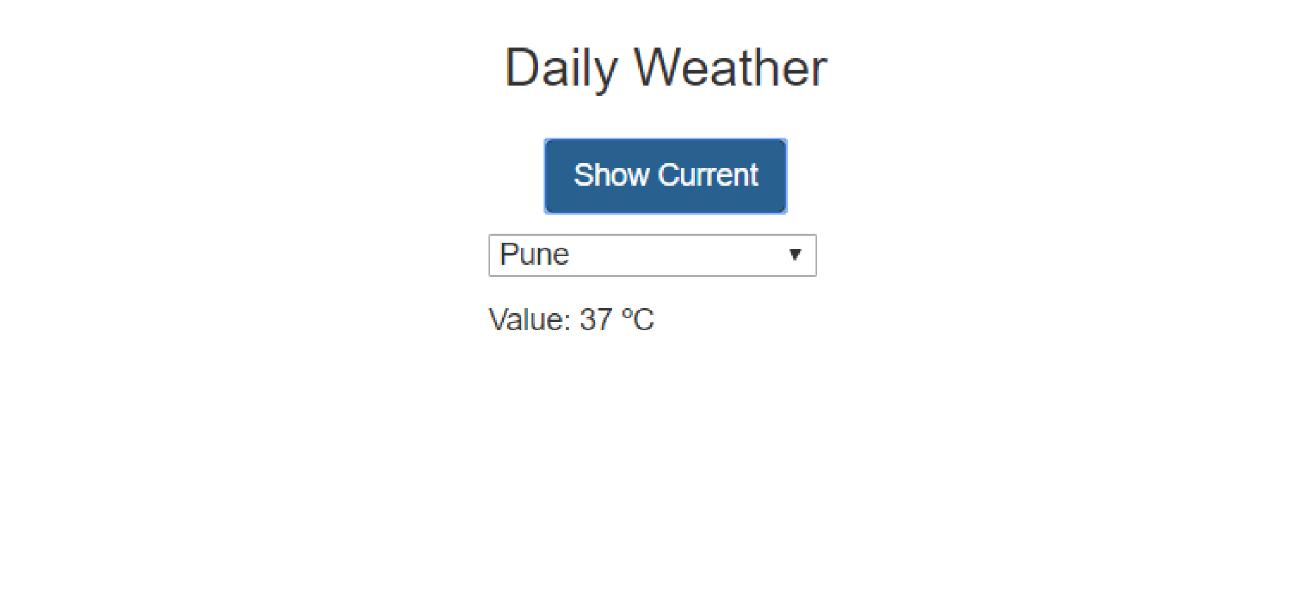
Need joins, transactions, frequent table scans?

Want DB engine to manage DB integrity?

Team has SQL skills?

Amazon RDS

Task: Build a API based simple backend for Weather App

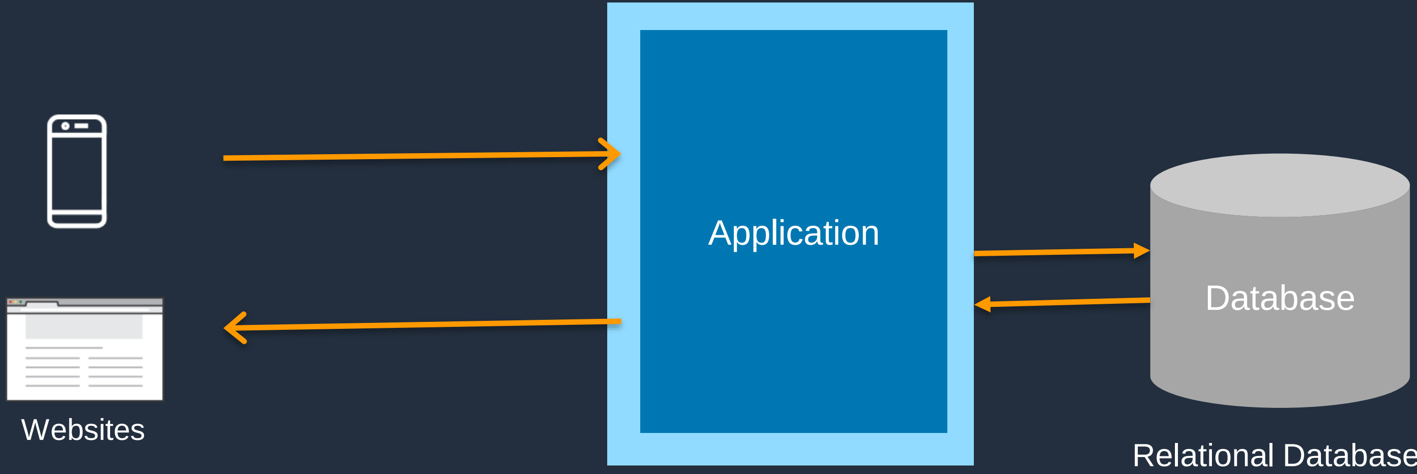


Daily Weather

Pune ▼

Value: 37 °C

Task: Build a API based simple backend for Weather App



Application

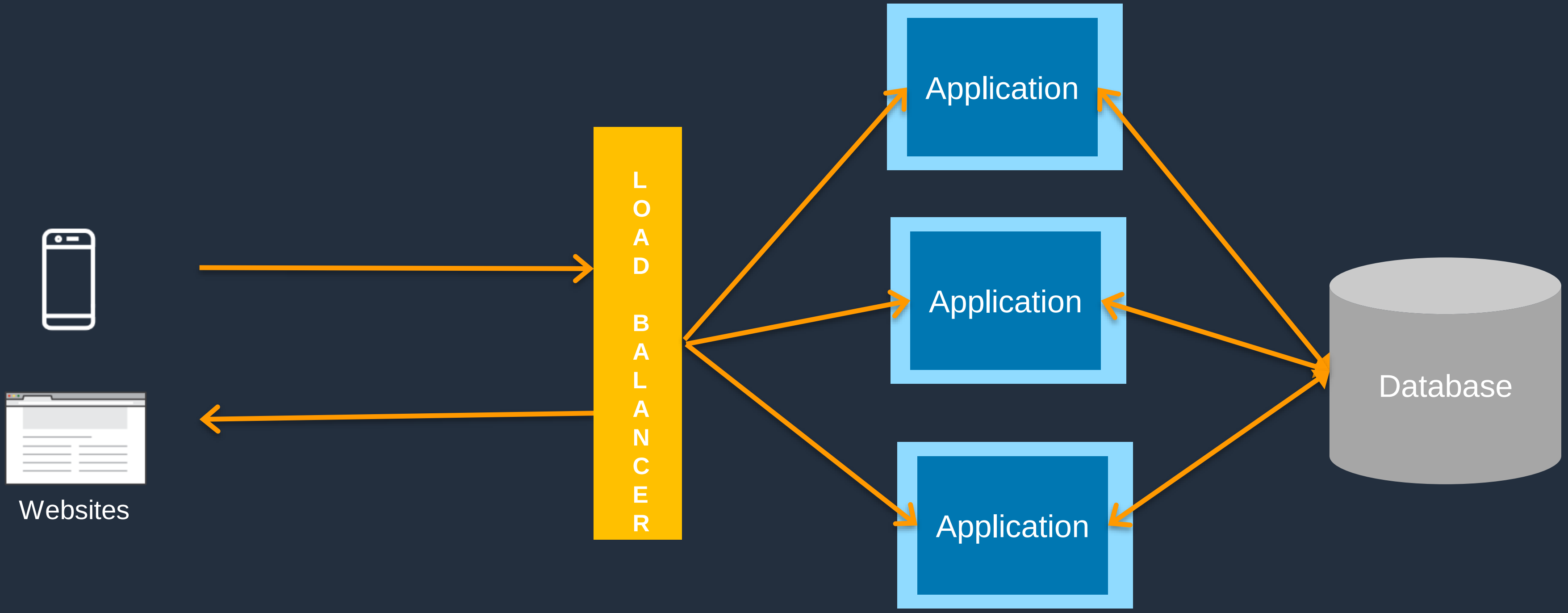
Database

Websites

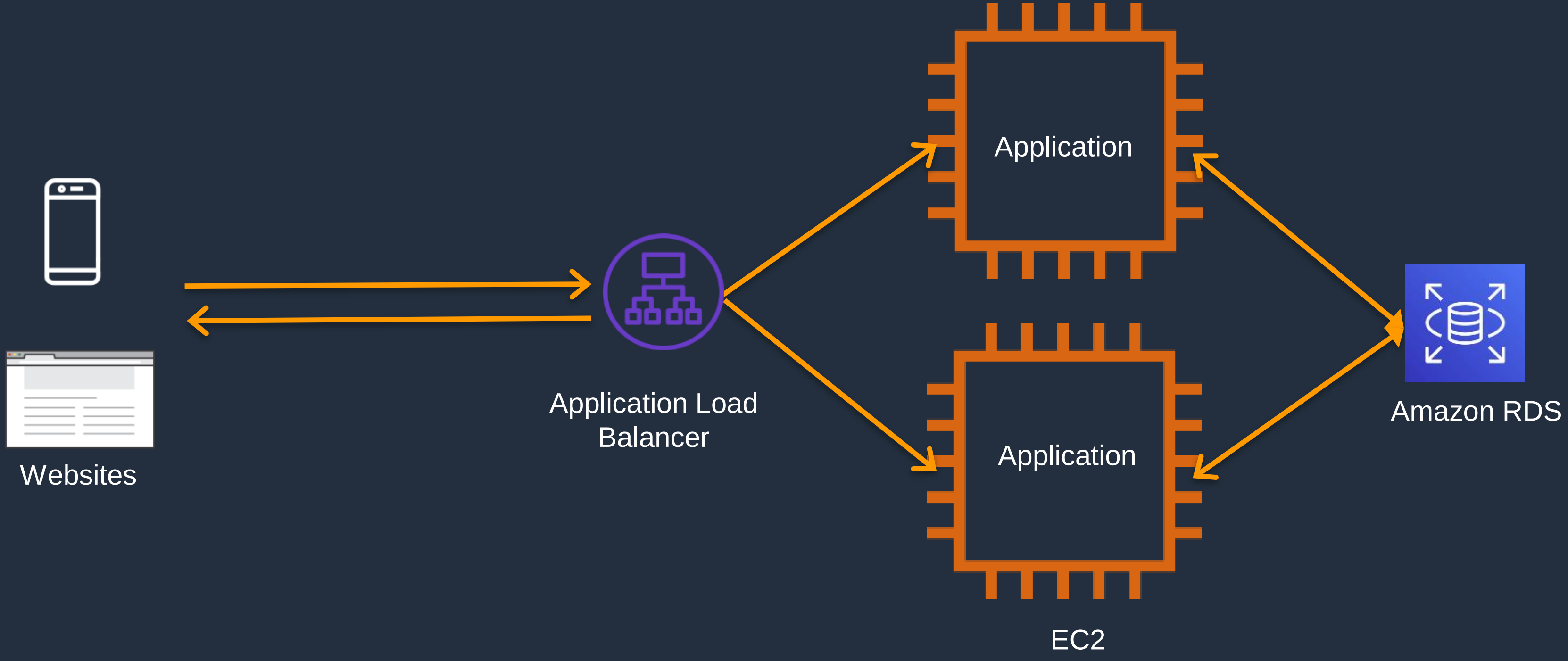
Relational Database

Python
Node.js
Java
C/C++
Go…

How do you scale your application?



Let’s host this on AWS using IaaS services (EC2)

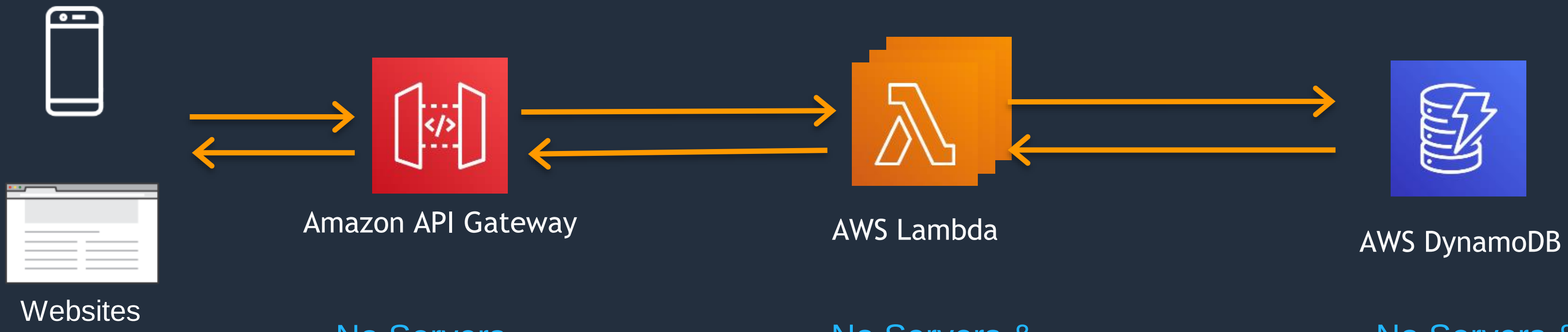


Do you see a problem with these architectures?

- Think about Scalability
- Think about Reliability
- Think about Performance
- Think about Operational efficiency
- Think about Cost

Solution: Let’s host this on AWS using Serverless services

http://d1osdxlszoytz4.cloudfront.net/



Websites

Amazon API Gateway

AWS Lambda

AWS DynamoDB

No Servers & Scales as per traffic

No Servers & concurrent executions based on API requests

No Servers & Scales based on RCU and WRU configurations

Introducing Amazon ElastiCache

Fully-managed, Redis or Memcached compatible, low-latency, in-memory data store

Extreme Performance

In-memory data store and cache for sub-millisecond response times

Fully Managed

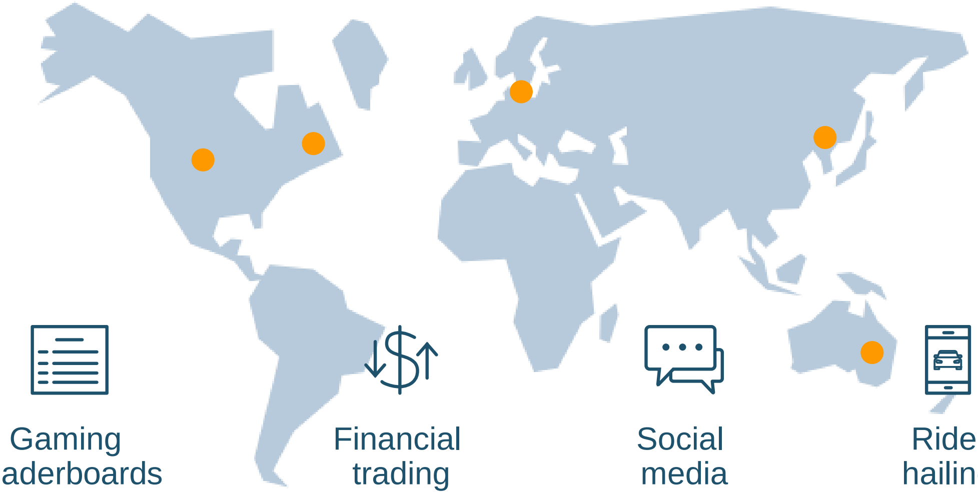
AWS manages all hardware and software setup, configuration, monitoring

Easily Scalable

Read scaling with replicas. Write and memory scaling with sharding. Non disruptive scaling

µs is the new ***ms***

Internet-scale apps need low latency and high concurrency



Gaming leaderboards

Financial trading

Social media

Ride hailing

Dating

Media streaming

Session stores

Users	1M+
Data volume	TB-PB-EB
Locality	Global
Performance	Milliseconds to microseconds
Request Rate	Millions
Access	Mobile, IoT, Devices
Scale	Up-Out-In
Economics	Pay as you go
Developer access	Instant API access

Developers use various approaches to reduce latency

In-memory databases and data grids

Specialized hardware such as multi-core processors, GPUs, accelerators

Data reduction approaches such as sampling, aggregation

Amazon ElastiCache

- In-memory cache in the cloud
- Improve latency and throughput for read-heavy workloads
- Supports open-source caching engines
  - Memcached
  - Redis
- Fully managed
- Multi-AZ

Examples

- Caching of MySQL database query results
- Caching of post-processing results
- Caching of user session and frequently accessed data

ElastiCache Redis

#1 Key-Value Store*

Fast in-memory data store in the cloud. Use as a database, cache, message broker, queue

Fully Managed & Hardened

AWS manages hardware, software, setup, configuration, monitoring, failure recovery, and backups

Secure & Compliant

VPC for cluster isolation, encryption at rest/transit, HIPAA compliance

Highly Available & Reliable

Read replicas, multiple primaries, multi-AZ with automatic failover

Easily Scalable

Cluster with up to 6.1 TiB of in-memory data

Read scaling with replicas

Write and memory scaling with sharding

Scale out or in

*: https://db-engines.com/en/ranking

ElastiCache Memcached

Fully Managed Memcached

Fast in-memory data store in the cloud. Use as a cache to reduce latency and improve throughput

Secure & Hardened

VPC for cluster isolation

Easily Scalable

Sharding to scale in-memory cache with up to 20 nodes and 8.14 TiB per cluster

Amazon
Neptune

Fully managed graph database

Supports open graph APIs

Scalable

ACID compliant

Multi-AZ

Amazon Neptune

Fully managed graph database for highly connected data

Open

Supports Apache TinkerPop™ & W3C RDF graph models

Fast & Scalable

Store billions of relationships; query with millisecond latency

Reliable

6 replicas of your data across 3 AZs with full backup and restore

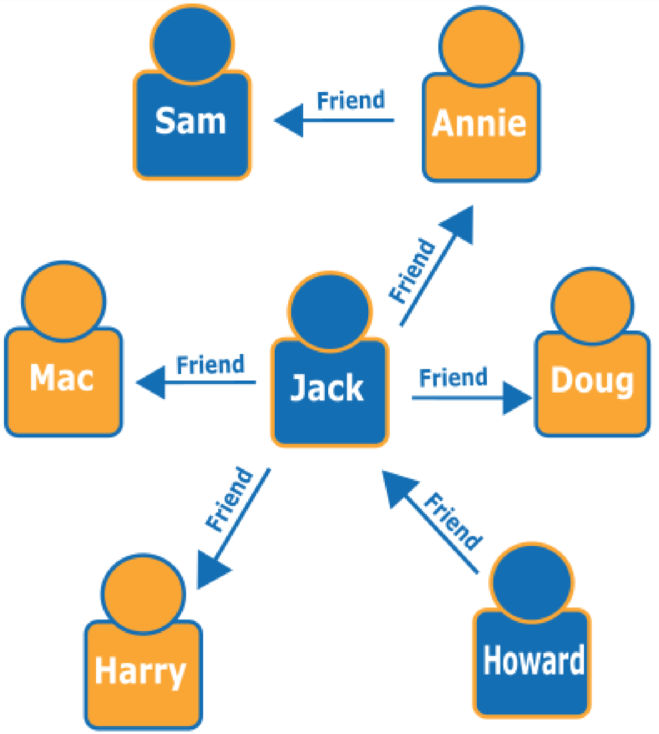
Easy

Gremlin
SPARQL

Build powerful queries easily with Gremlin and SPARQL
+
GRAPHQL with AppSync

Use cases for highly connected data

- Social networking
- Recommendations
- Knowledge graphs
- Fraud detection
- Life sciences
- Network and IT operations



Preview

Amazon Timestream

Fully managed time series database

1,000x faster at 1/10th the cost

Built-in analytics

Serverless

Building with time-series data is challenging

Relational databases

- Unnatural for time-series data
- Inefficient time-series data processing
- Rigid schema inflexible for fast moving time-series data

Existing time-series databases

- Difficult to scale
- Difficult to maintain high availability
- Limited data lifecycle management

Amazon Timestream

(Preview)

1,000x faster at 1/10th the cost of relational databases

Collect fast moving time-series data from multiple sources at the rate of millions of inserts per second

Trillions of daily events

Capable of processing trillions of events daily; the adaptive query processing engine maintains steady, predictable performance

Analytics optimized for time series data

Built-in analytics for interpolation, smoothing, and approximation to identify trends, patterns, and anomalies

Serverless

No servers to manage; time-consuming tasks such as hardware provisioning, software patching, setup, & configuration done for you

Amazon QLDB

Fully managed ledger database

Immutable and transparent

Cryptographically verifiable

Scalable

Serverless

Amazon Quantum Ledger Database (QLDB)

Fully managed ledger database

Track and verify history of all changes made to your application’s data

Immutable

Maintains a sequenced record of all changes to your data, which cannot be deleted or modified; you have the ability to query and analyze the full history

Cryptographically verifiable

Uses cryptography to generate a secure output file of your data’s history

Highly scalable

Executes 2–3X as many transactions as ledgers in common blockchain frameworks

Easy to use

Easy to use, letting you use familiar database capabilities like SQL APIs for querying the data

Common customer use cases

Banking & Finance

Keeping track of transactions, trades and accounts

E-Commerce

Where’s my stuff?

Transport & Logistics

Tracking transportation of goods

HR & Payroll

Tracking changes to an individual’s profile

Manufacturing

Recording components used in manufacturing

Government

Tracking vehicle title history

(Now GA)

Amazon DocumentDB

Fully managed document database

MongoDB-compatible

Multi-AZ

Performance at scale

Why use a document database?

The JSON document model maps naturally to application data

Each document can have a different data structure and is independent of other documents

Index on any key in a document, and run ad hoc and aggregation queries across your data set

Use cases for document databases

Content Management

Mobile

Personalization

Catalog

Retail and Marketing

User profiles

Use case: Profile Management

users table

id	username	first_name	last_name
181276	sue1942	Susan	Benoit

tankfight_users table

id	hi_score	global_rank
181276	3185400	5139

```
{
        id: 181276,
        username: 'sue1942',
        name: {first: 'Susan',
                        last: 'Benoit'},
}
```

```
{
        id: 181276,
        username: 'sue1942',
        name: {first: 'Susan',
                        last: 'Benoit'},
        tankfight: {
                hi_score: 3185400,
                        global_rank: 5139
        }
}
```

It’s all about

choice

Performance-oriented
Cost-oriented

Common data categories and use cases

Relational	Key-value	Document	In-memory	Graph	Time-series	Ledger	Warehouse
Referential integrity, ACID transactions, schema-on-write	High throughput, low-latency reads and writes, endless scale	Store JSON documents with quick access, query on any attribute	Query by key with microsecond latency	Quickly and easily create and navigate relationships between data	Collect, store, and process data sequenced by time	Complete, immutable, and verifiable history of all changes to application data	High performance querying on large volumes of data
RDS	DynamoDB	DocumentDB	ElastiCache Redis Memcached	Neptune	Timestream	QLDB	Redshift

Thank you!

Chetan Agrawal – agrcheta@amazon.com