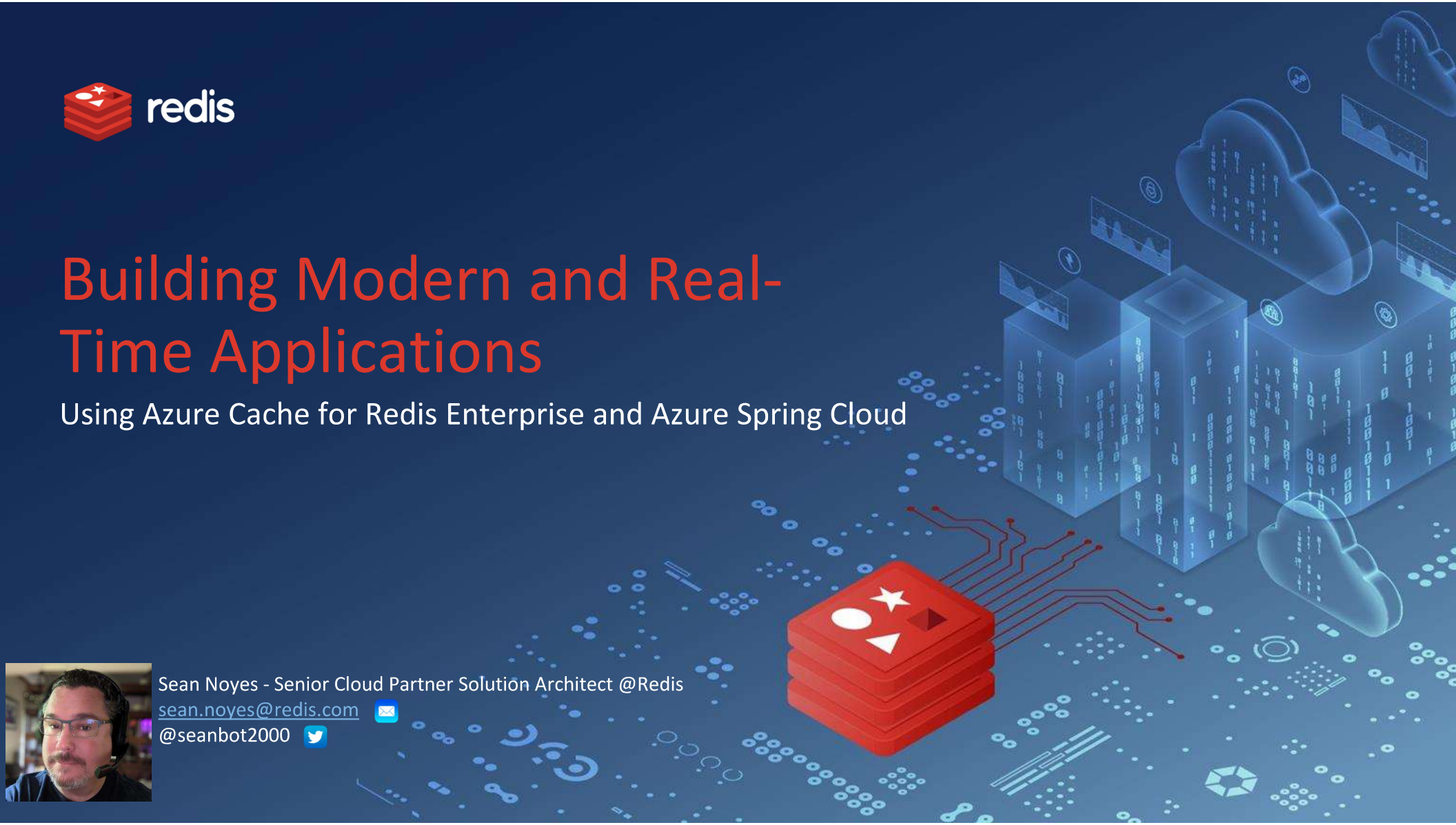


# Building Modern and Real-Time Applications

## Using Azure Cache for Redis Enterprise and Azure Spring Cloud



Sean Noyes - Senior Cloud Partner Solution Architect @Redis
sean.noyes@redis.com
@seanbot2000

# Agenda

## Today's topics

- A short history of the NoSQL movement
- Modern application architecture and Redis Use Cases
- Azure Cache for Redis Enterprise
- An example of modern data-centric architecture
- Let's build an environment and deploy something!
- Q&A

# A short history of the NoSQL movement

# A short history of the NoSQL movement

## What's a NoSQL anyway?

- NoSQL today stands for 'Not only SQL'
- Whereby SQL is used as a synonym for Relational Database Management Systems (RDBMS)
- First coined in 1998 by Carlo Strozzi, who built an open source relational database without a SQL interface
- It's a movement, not a standard (unlike SQL, which *is* a standard)

# A short history of the NoSQL movement

## Why the need for non-relational databases?

**Not-Only SQL**

- Optimized for specific use cases
- Flexible schemas
  - Can change on-demand, logic moved into app tier
- Scalability & performance
  - Scale out instead of scale up
- Availability
  - Designed from the ground up to be distributed and always available……but at the sacrifice of…..
- Eventual consistency
  - With varying levels of strength, usually not ACID compliant
- Cloud Native from the ground up
  - Born in the modern age
- Simplified administration
  - Ops/DevOps instead of DBAs

**SQL (RDBMS)**

- Aim to be general purpose systems
  - Often good enough at most cases, but not very well optimised for specific ones
- Fixed schemas
  - Changes often mean migrations
- Typically hard to scale (but might be good enough)
  - Favoring a scale-up over a scale-out approach
  - Which limits scalability
  - And availability at scale, esp. geographically
- But they are consistent (ACID compliance)
- Complex administration
  - Need for DBAs
- Though there are a lot of new cool kids on the block!
  - Amazon RDS, Aurora, Azure SQL
  - Yugabyte

# A short history of the NoSQL movement

## Specific use cases for NoSQL

- Key-value store
  - Memcached
- Data structure store
  - Redis
- Wide-table store
  - Cassandra
- Document database
  - MongoDB, Cosmos DB
- Graph database
  - Neo4J
- Time series database
  - InfluxDB



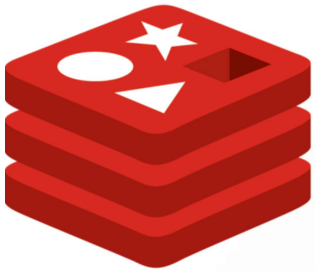


# Redis in a nutshell

*Key-value and data structure store*

*NoSQL Database, Cache, Message Broker*

*In-Memory Architecture*



redis

**Re**mote **Di**ctionary **S**erver

Open-Source (BSD License)

*Written in C*

*Single Threaded*

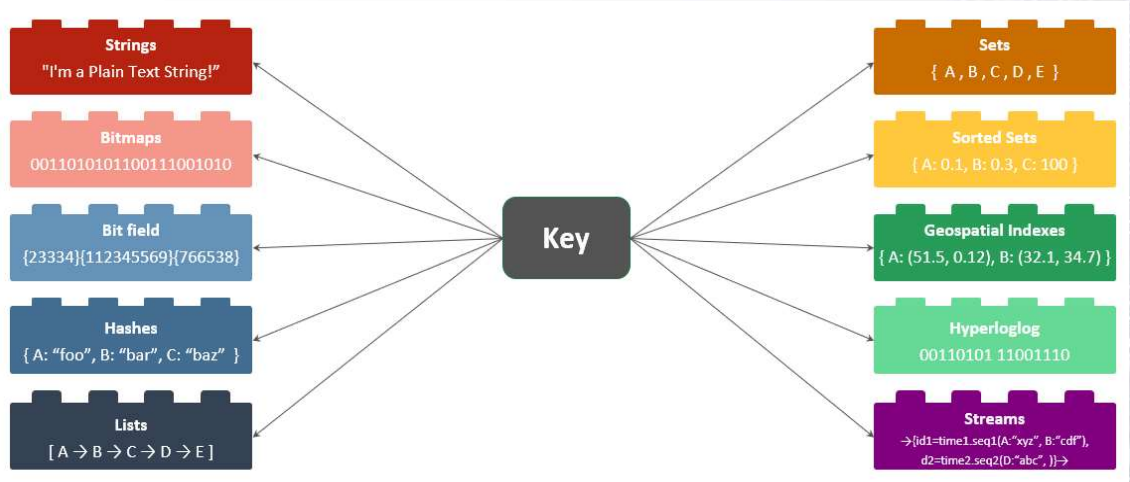
*Multiple Persistence Options*

*ACID Compliant*

# The core principles of Redis

## Three pillars

- Performance
  - All data in-memory
  - Single-threaded, lock-free
  - Most commands O(1) complexity
  - Efficient communication protocols
- Simplicity
  - Key -> Data structure
  - Data structure = Strings, (Sorted) Sets, Lists, Geospatial indexes, Hashes, Bit fields/maps, Pub/Sub, HyperLogLog and Streams
  - Extensible via modules, e.g. time series, graphs, search, etc.
- Use-case driven design
  - Data structures map to specific use cases, e.g. caching, user sessions, location services, recommendations, leaderboards, etc.



# Modern, real-time data architectures (with Azure Cache for Redis)

USE CASE 1

# Distributed cache

## Use case examples

Handling spikes in traffic during major events

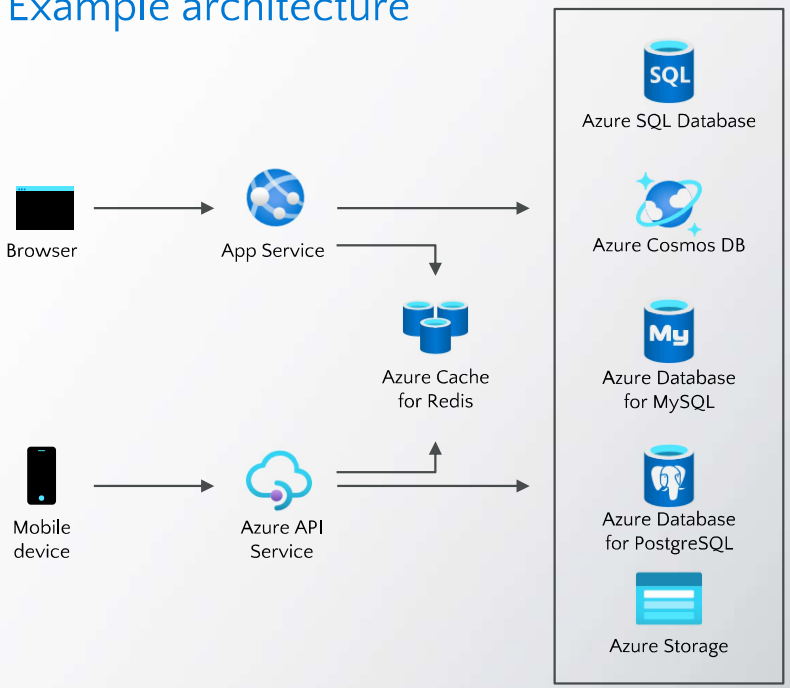
Serving commonly accessed data to users, like inventory or pricing data

Reducing compute load on a database

Placing content geographically closer to a consumer

Output caching

## Example architecture




redis

USE CASE 2

# Session store

## Use case examples

eCommerce shopping carts

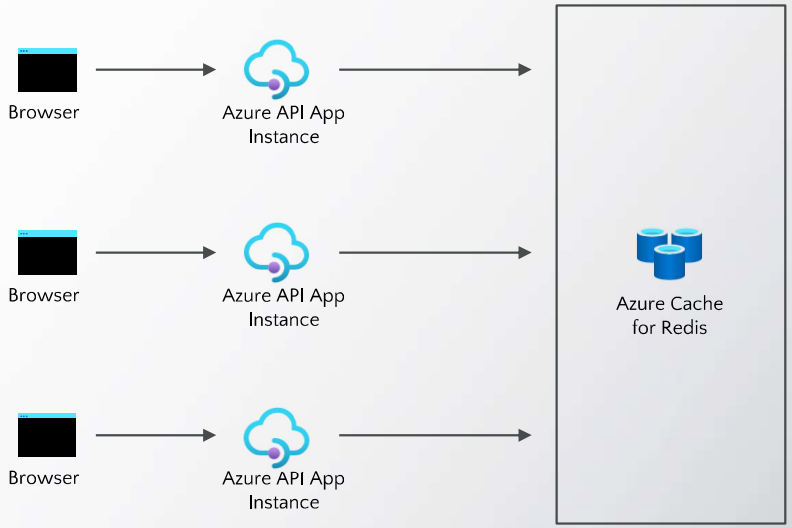
Storing user cookies

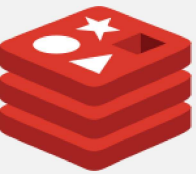
User login and session state

Customized pricing quotes

IoT telemetry

## Example architecture



redis

USE CASE 3

# Pub/Sub or Message Broker

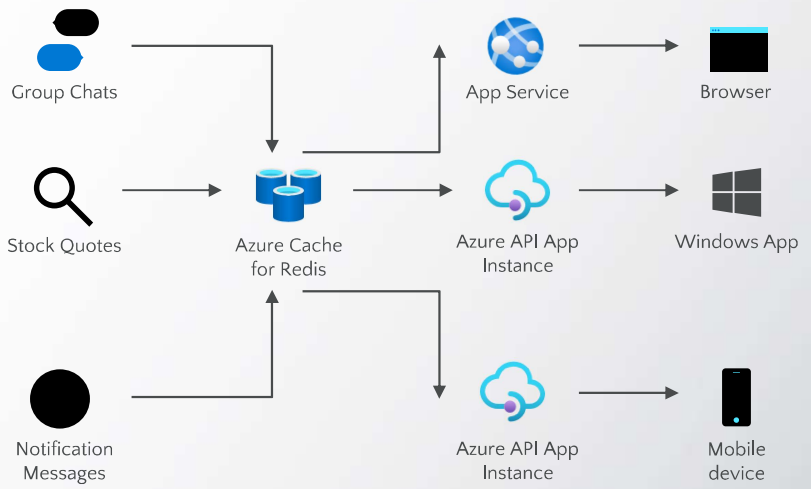
## Use case examples

Publishing news, financial data, or application updates to users

Handling chat messages

Communication between microservices

## Example architecture




USE CASE 4

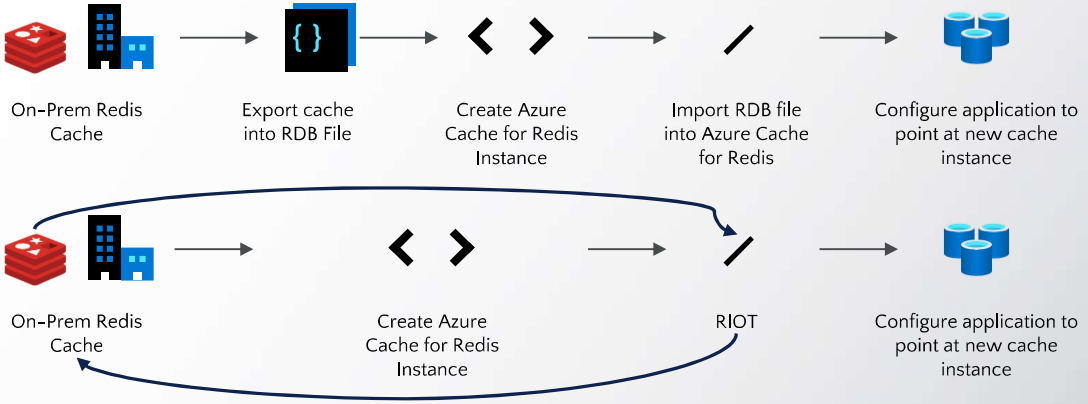
# Migrate Redis to the cloud

## Use case examples

Migrating application from on-premises to the cloud

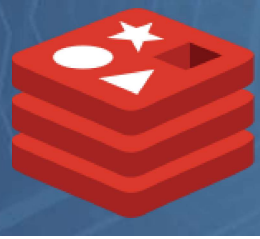
Modernizing IaaS application through new PaaS services

## Example architecture




redis

# Azure Cache for Redis Enterprise



redis

# Azure Cache for Redis Offerings

## Basic, Standard, and Premium Tiers

Based on open-source Redis with core features like replication, clustering, and network isolation.

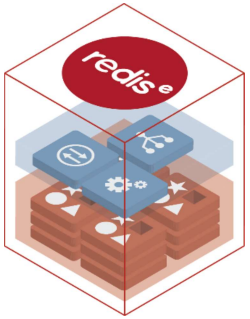
**Best for**: general purpose and dev/test usage

## Enterprise, and Enterprise Flash Tiers


Based on Redis Enterprise with advanced features like active geo-replication, 99.99%+ availability, and Redis Modules

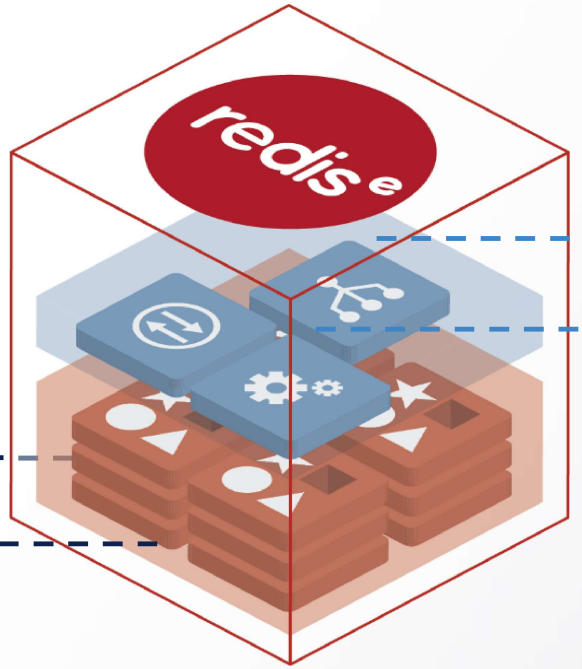
**Best for**: enterprise-scale and advanced usage

# Redis Enterprise Terminology




- **Redis Instance** – single-threaded Redis OSS database
  - Commonly referred to as a **shard** with either a role as a master or slave
- **Redis Enterprise Database** – logical entity that manages your entire dataset across multiple Redis Instances
  - not to be confused with the OSS database inside every Redis instance
  - multi-tenant, maximizing infrastructure utilization (reduces overall TCO)
- **Redis Enterprise Nodes** – physical servers, virtual machines, containers, and/or cloud instances
  - on which the Redis Enterprise installation package is installed and configured
  - each node is a container capable of running multiple Redis Instances (i.e., **shards**)
- **Redis Enterprise Cluster** – set of Redis Enterprise nodes pooling resources
  - allows you to create any number of Redis Enterprise nodes to scale up/out

# Redis Enterprise Node




**Cluster Management**

Govern Cluster, Orchestrate Failure Detection, Failover, Stats Collection & more

**Secure UI & REST API**

Allow programmable and visual administration over HTTPS

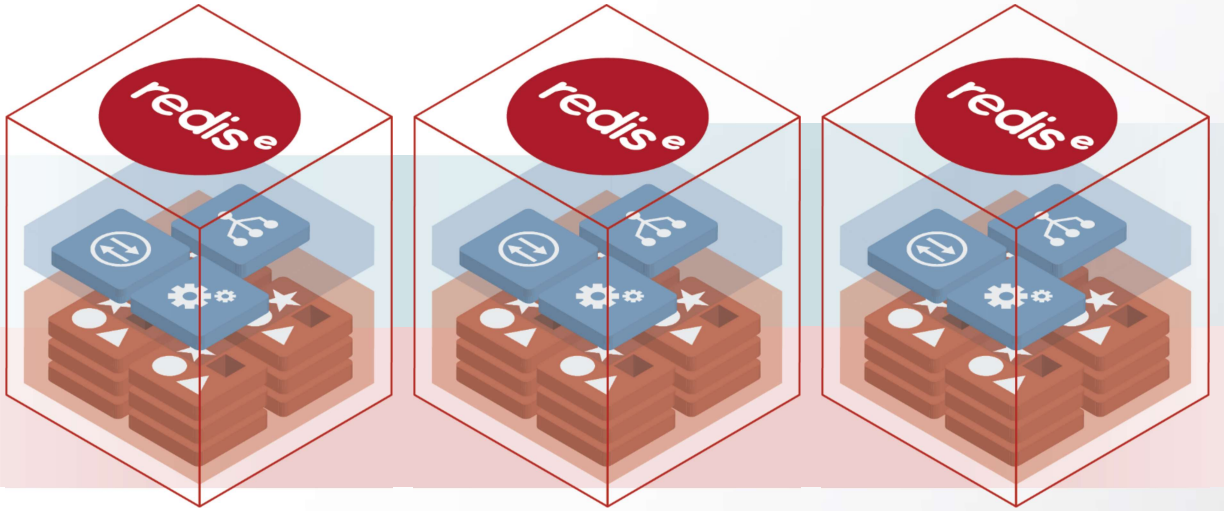
**Redis Instances**

*Based on Open Source Redis*

**Proxy**

*Scale Connections*
*Improve Application Performance*

# Shared-Nothing Symmetric Architecture




Cluster Management Path

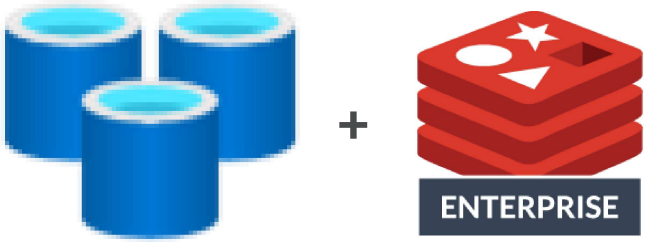
Data Path

Node Watchdog
Cluster Watchdog

Redis Shards
Proxies
Database discovery

Node 1 | Node 2 | Node N (odd number)

# Azure Cache for Redis, Enterprise Tiers




Best of both worlds:

The advanced functionality of Redis Enterprise plus the management and integration of Azure Cache for Redis

## Powerful Features

- Active-Geo Replication
- RediSearch
- RedisBloom
- RedisTimeSeries
- Redis on Flash
- Up to 99.999% availability
- Redis 6.0
- Azure Private Link

## Joint Support

Microsoft provides round-the-clock support and takes the first call.

Direct line to experts at Redis Labs for escalations.

## Streamlined Administration

- Integrated billing
- Azure spend commitments
- Azure security integration
- Azure monitoring tools
- Configuration through the Azure Portal, CLI, ARM, and Terraform

# Redis Enterprise

## Redis on Flash

Running Redis on high speed NVMe flash storage

Extends your DRAM capacity with SSD and persistent memory

10X larger cache sizes at a lower price per GB

# RedisBloom Module

## Extremely efficient probabilistic data structures

### Features

- Includes data structures that trade perfect accuracy for memory efficiency and low latency:
  - **Bloom and Cukoo filters**: determine if an item is definitely not in a set, or probably is in a set.
  - **Count-min sketch**: determine the frequency of events in a stream.
  - **Top-K**: Find the *k* most frequently seen items

### Use-cases

- Checking if usernames are taken
- De-duplication
- Detecting transaction patterns
- Monitoring/alerts
- Leaderboards

### Applications

- Fraud detection
- User management
- Advertising

# RediSearch Module

Real-time secondary index and search engine, built into Redis

## Features

- Extremely fast
- Incremental indexing without performance loss
- Advanced search functionality
  - Prefix
  - Fuzzy
  - Phonetic
  - Stemming
  - Synonyms
  - Autocomplete
  - Geo indexing and filtering

## Use-cases

- Secondary Index
- Full-text search
- Query engine

## Applications

- Enterprise Search
- Real-time inventory
- Customer service & insights
- External database indexing

# RedisTimeSeries Module

## High-throughput time-series database

### Features

- Handles extremely high volume of data inserts
- Downsampling/compaction
- Time-based queries (e.g., start time & end time)
- Configurable retention period
- Labels for secondary indexing
- Aggregated Queries
  - Min, Max, Avg, Sum Range, Count, First, Last
  - STD.P, STD.S, Var.P, Var. S

### Use-cases

- Time-series database
- High-throughput data ingestion

### Applications

- IoT telemetry
- Application monitoring
- Anomaly detection

# Active Geo-Replication

Simultaneous read & write operations on multiple globally distributed instances

Worldwide scale with the same local latency

Greater resiliency and data consistency across the globe

redis

# Industry-Leading Availability

99.9%

## Default

Built-in replication with automatic failover to replica node

99.99%

## Multi-AZ

Placing replicas in different availability zones

Up to

99.999%

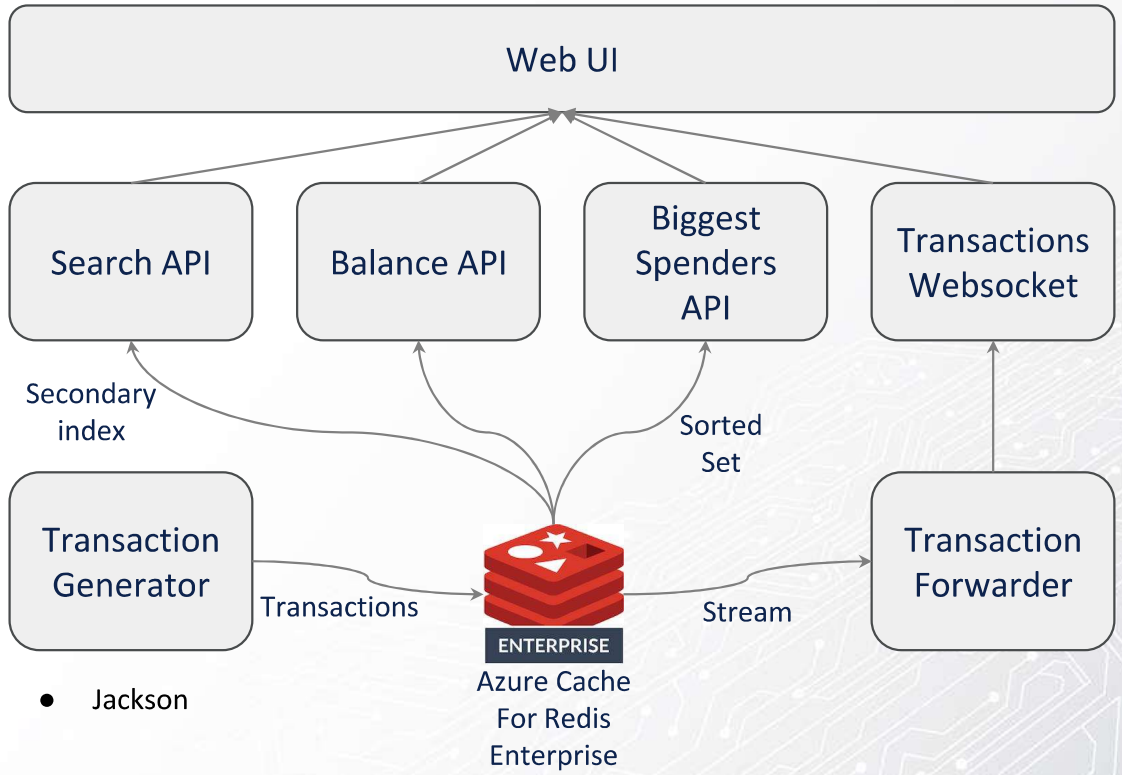
## Active Geo-Replication

Adding multi-primary writes across regions

redis

# Let’s take a look at an actual app!

# Application architecture




- Vue.js
- ApexCharts.js
- HTML
- Bootstrap

- Spring MVC
- Spring Redis RediSearch
- Websocket
- Spring Security
- Spring Session

- Spring Data Redis Reactive
- Lettuce (TS)

Microsoft Azure

Azure Spring Cloud

# Code, cloud, cache, and more!

# Thank You