Configuring Cache and Expiration Policies for Azure Redis Cache

James Millar
FREELANCE SOFTWARE DEVELOPER

@jamesmillar www.james-millar.co.uk

Azure Redis Cache Pricing Tiers

Basic

Minimal feature set
No SLA
Development and test

Standard

2 Replicated nodes
99.9% availability
53 GB of memory
20,000 clients

Premium

Redis cluster
Low latency
99.95% availability
40,000 clients

Azure Redis Cache Pricing Tiers

Enterprise

Full Redis feature set

99.99% availability

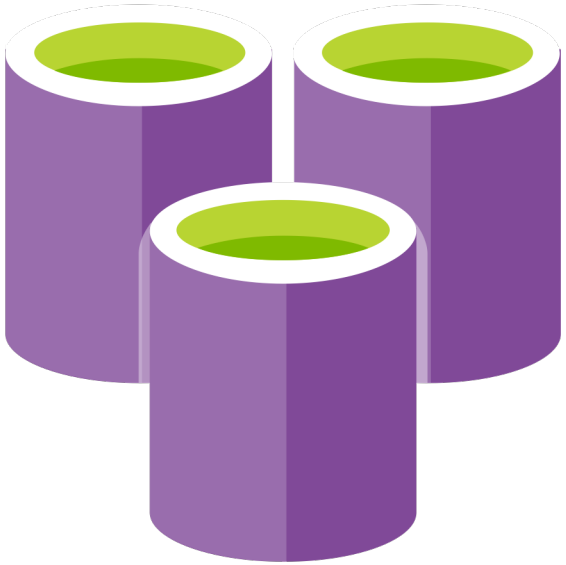
Enterprise Flash

Fast non-volatile storage

You can scale up, but you cannot scale down!

Understanding Default Caching Behaviour

Understand Caching

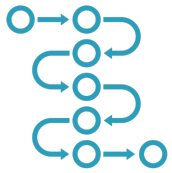


Improve performance and scalability

Move frequently accessed data closer

Faster response times

When Should We Cache?



Repeatedly accessed data

Data source performance

Data contention

Physical location

Managing Lifetime in Redis Cache

No default expiration

Content exists until it’s removed

Must set TTL manually

Calculating Cache Duration

Rate of Change

Long expiry for static data

Short expiry for volatile data

Risk of Outdated Data

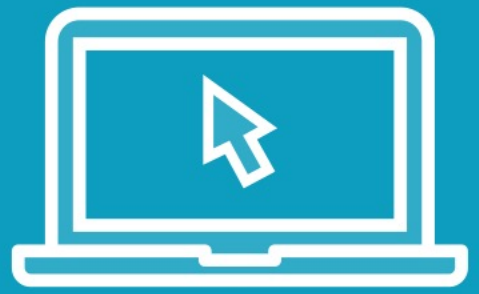
Lower TTL to match data change

```
_cache.StringSet("myKey", "my Value", new TimeSpan(3, 0, 0));
```

Setting Expiration Times for Redis Cache

By default, values will not expire

Demo



Configuring Redis Cache Expiration

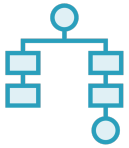
Redis Cache Best Practices

Best Practices

Watch out for data loss

Set expiry times to manage content lifetime

Add jitter to spread database load



Avoid caching large objects

Host Redis in the same region as your application

Up Next:
Implementing Application Caching Patterns