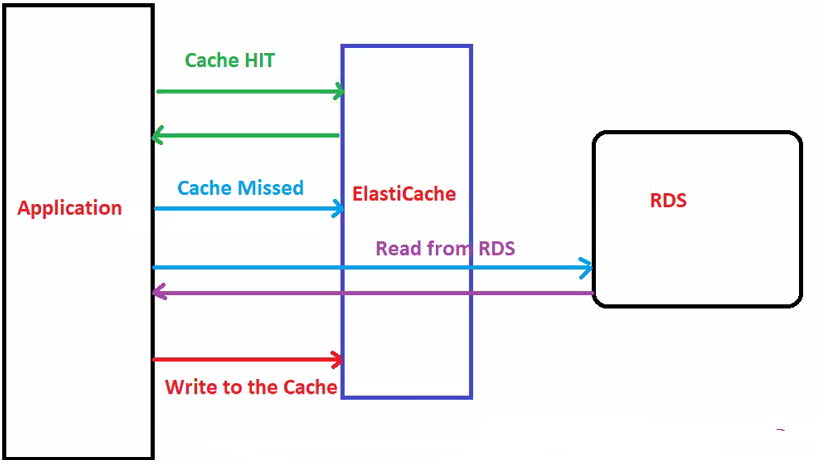
**ElastiCache**

* The same way RDS is to get managed Relation Databases
* ElastiCache is web service (PaaS) that makes it easy to setup, manage, and scale a distributed **in-memory data store or cache** environment in AWS Cloud
* Caching is a technique to store frequently accessed information in a temporary memory location on a server. Read-intensive web applications are the best use-case candidates for a cache service.
* **There are a number of caching servers used across applications, the most notable are Memcached, Redis, and Varnish**
* It provides **high-performance, scalable**, and cost-effective caching solution
* It removes the complexity of deploying and managing a distributed cache environment
* It improves latency and throughput for read-heavy application workloads
* It improves application performance by storing critical data in memory for low-latency access
* ElastiCache is to get managed Radis or Memcached
* ElastiCache is compliant with **Memcached and Radis**
* ElastiCache support Multi AZ with Failover Capability
* **ElastiCache mostly used for read purpose**
* Helps reduce load off of databases for read intensive workloads
* Write Scaling using Sharding
* Read Scaling using read Replicas
* **If you want to increase the performance, choose ElastiCache**
* **ElastiCache mainly used for read purpose we can use write purpose also but mainly read only**
* AWS takes care of OS maintenance/Patching, optimizations, setup, configurations, monitoring, failure recovery and backups



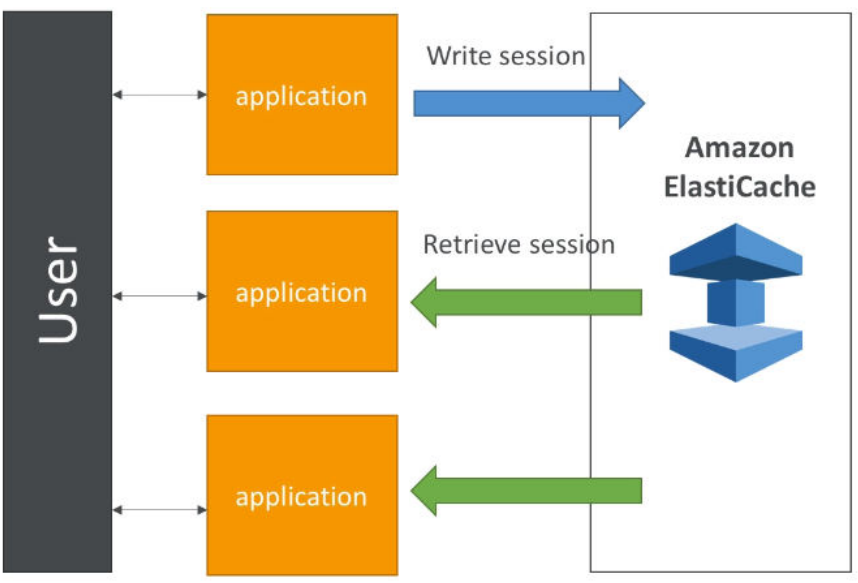
* ElastiCache is a web service that makes it easy to deploy, Operate and scale an in-memory cache in the cloud.
* The service improves the performance of web applications by allowing you to retrieve information from fast, managed,

in-memory caches, instead of relying entirely on slower disk-based database.

* ElastiCache can be used to significantly improve latency and throughput for many read-heavy application workloads
* Caching improves application performance by storing critical pieces of data in memory for low-latency access.
* Cached information may include the results of I/O-intensive database queries or the results of computationally intensive calculations.
* ElastiCache is a good choice if your database is particularly read heavy and not prone to frequent changing.

**ElastiCache also used for storing users’ session data**

* User logs into any of the application
* The application writes the session data into ElastiCache
* The user hits another instance of our application
* The instance retrieves the data and the user is already logged in



**ElastiCache Nodes:**

* A node is the smallest building block of ElastiCache deployment It's fixed size chunk of network-attached RAM
* Each node runs an instance of the engine (**Radis or Memcached**), when you create a cluster
* ElastiCache for Radis and Memcached both does not support IAM Authentication

**Shard**

* **A collection of nodes / Servers is called “Shard”**
* **Collection of nodes/ Shards is called “Cluster”**

**Radis**

* Radis can be used as an **in-memory key-value data store**
* Provides fast, sub millisecond data performance
* Provides **high-availability and scalability**
* **It is purely act as a database/normal data base**
* **Both Redis and Memcached key-value store data base**
* Radis support **shard**
* Multi-AZ with automatic failover of failed primary cluster to read replica
* **Supports read-only replica nodes**

**Redis Clusters**

* A Redis (Cluster mode disabled) cluster will never have more than one shard
* A Redis (Cluster mode enabled) cluster can have from **1 to 90 shards**
* You can create a cluster with higher number of shards and lower

number of replicas totalling up to 90 nodes per cluster

* The node or shard limit can be increased to a maximum of 250 per

cluster

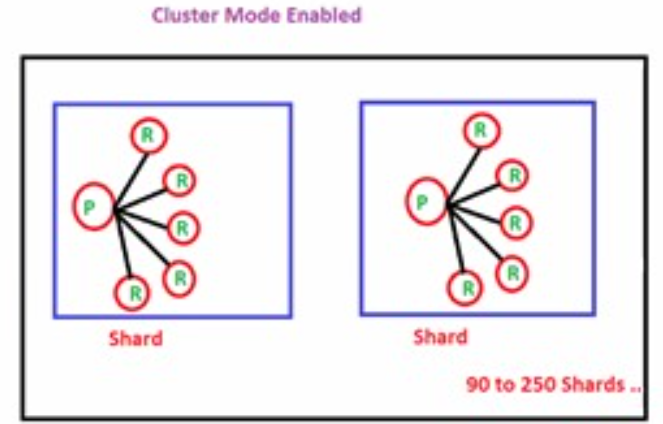
* **Each shard has 6 nodes** ***i.e(1 primary node and 5 replica nodes)***

**We can have 1 replica or 2 replicas but max is 5 replicas and this is completely managed by AWS**

**Write will be happening on primary and read will be replicas**

**(Note This is when Cluster mode is enabled), if disable only shard**

* **Data cached on Radis persists by default even after reboot**
* **Redis In-memory data structure store used as database, cache and message broker.**
* **ElastiCache for Redis offers Multi-AZ with Auto-Failover and enhanced robustness.**



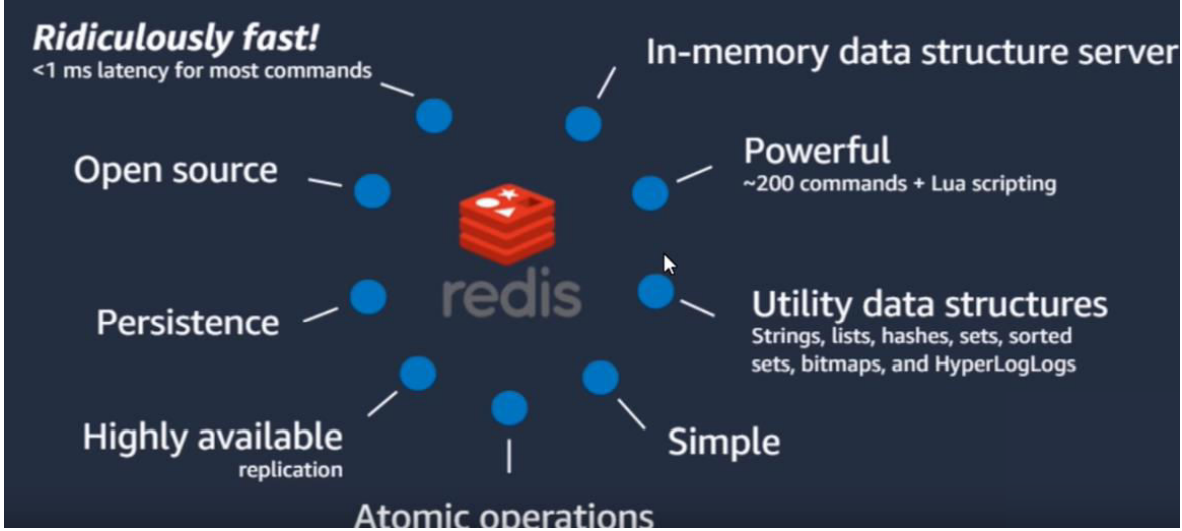
* A Redis Cluster (**Cluster made disabled**) always **contains just 1 shard**
* **1 primary and 5 replicas**
* **By default, Cluster mode will be disable**
* **Redis implements six fine-grained policies for purging old data, while Memcached uses the**
* **LRU (Least Recently Used) algorithm.**
* **Redis supports key names and values up to 512 MB, whereas Memcached supports only 1**
* **MB.**
* **Redis uses a HashMap to store objects whereas Memcached uses serialized strings.**
* **Redis provides a persistence layer and supports complex types like hashes, lists (ordered collections, meant for queue), sets (unordered collections of non-repeating values), or sorted sets (ordered/ranked collections of non-repeating values).**
* **Redis is used for built-in pub/sub, transactions (with optimistic locking), and Lua scripting.**
* **Redis 3.0 supports clustering.**
* **Multi AZ with Automatic Failover for DR if you don’t want to lose your cache data**
* **Super low latency (sub ms)**

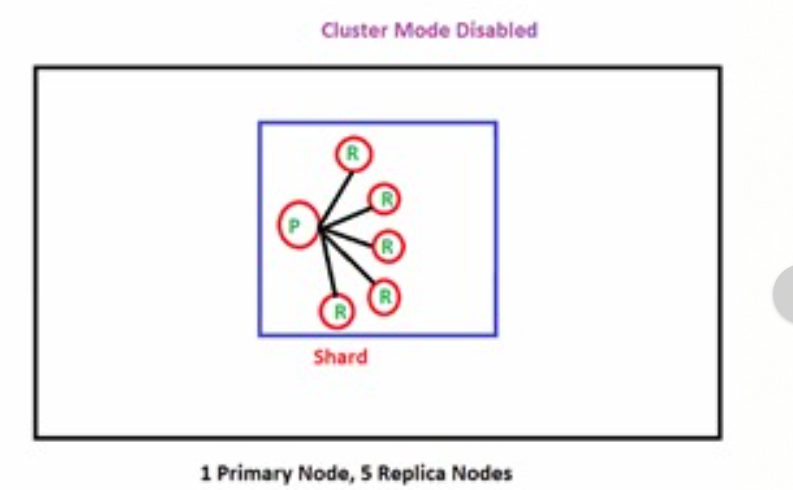
**Encryption**

**In-Transit: en**crypt all communications between clients and Radis server as well as between nodes

**At-Rest:** encrypt backups on disk and Amazon S3

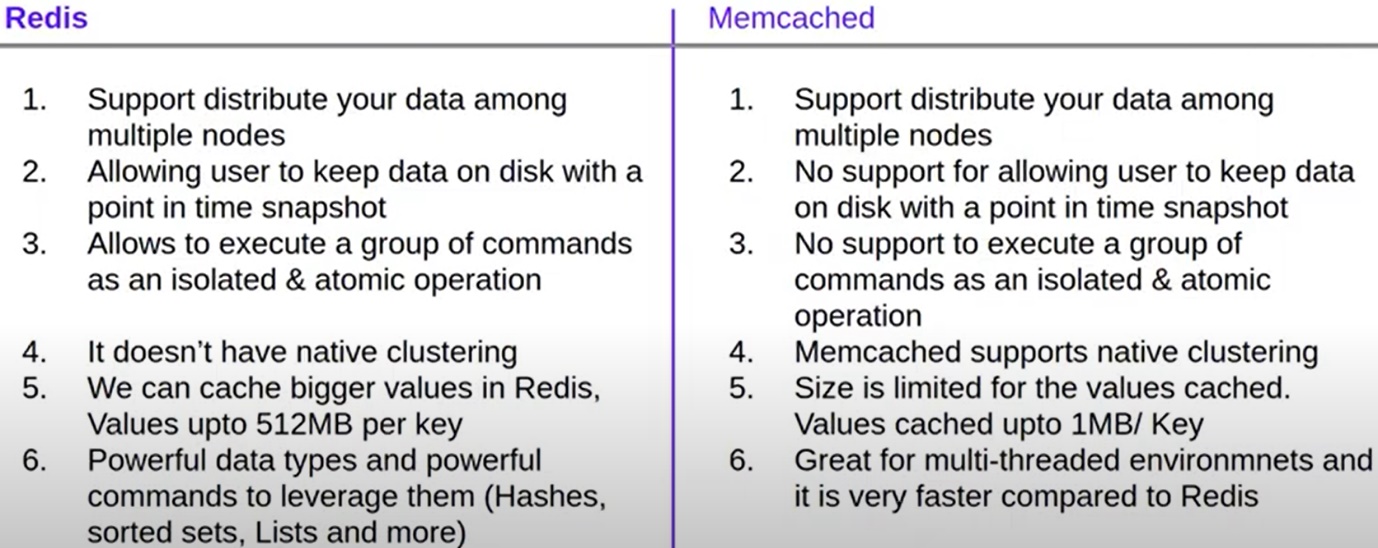
**Fully managed:** setup via API or console, automatic issuance and renewal

****



**Memcached**

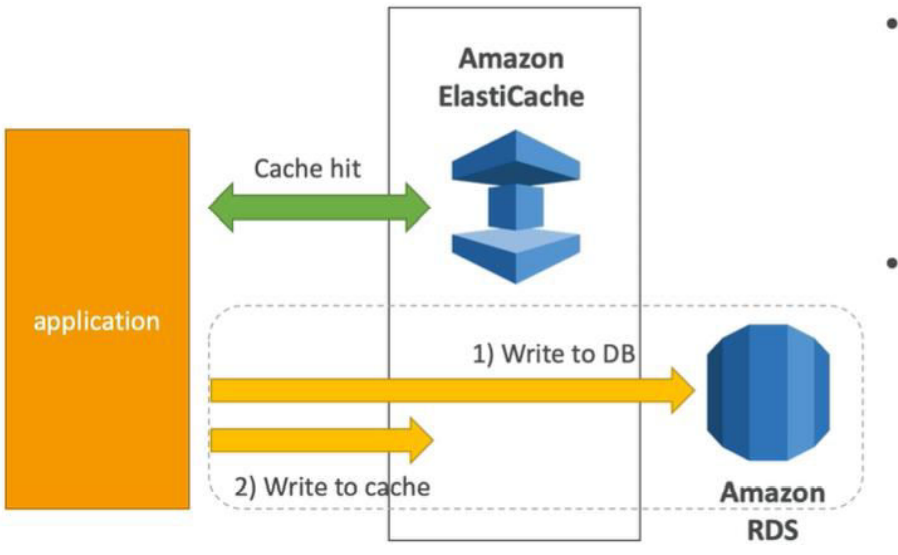
* Memcached-compatible **in-memory key-value store service**
* Can be used as cache or a data store
* it's fully managed, scalable and secure
* Provides fast, sub milliseconds data performance
* **Data cached on Memcached does not persist after reboot**
* It does not support HA, failover and backups
* Redis is more popular than Memcached
* **Memcached maximum 20 nodes**
* **Memcached is a High-performance, distributed memory object caching system,**
* **intended for use in speeding up dynamic web applications.**
* **If you need to scale horizontally use Memcached.**



**Mainly 2 strategies**

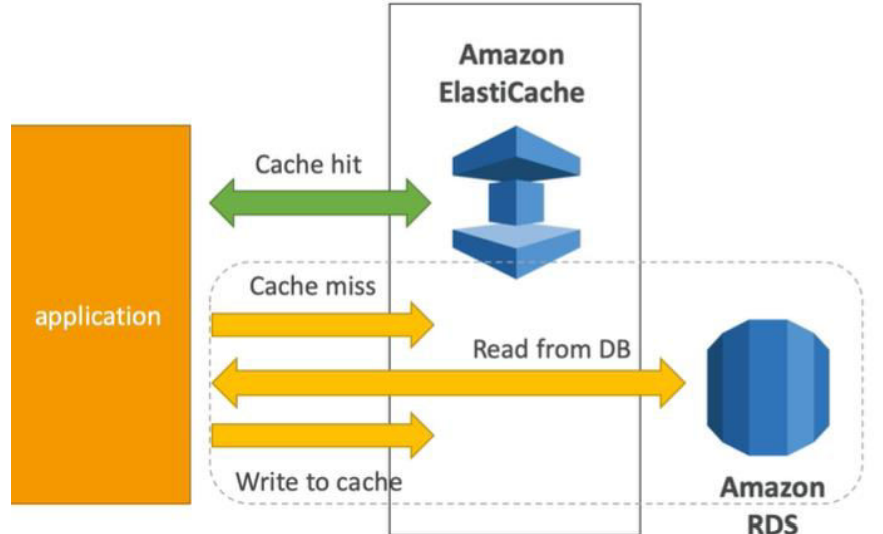
**1.Write Through**

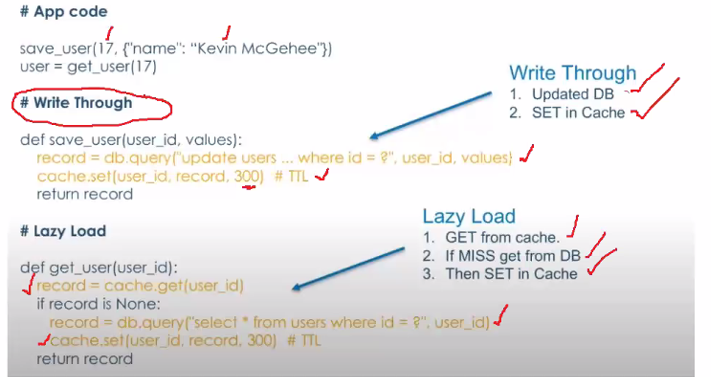
* Parallel Writes on Database and Cache
* Add or update when database is update

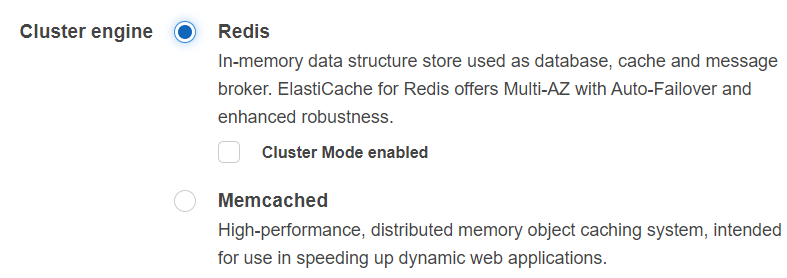


**2.Lazy Load: Load only when necessary**

* First check Cache
* if yes **OK**
* if no, get from Database
* and SET Cache







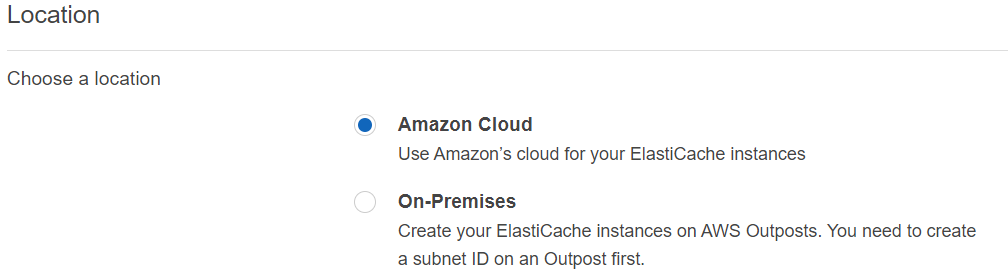
Live this section uncheck i.e Cluster **Mode enable**

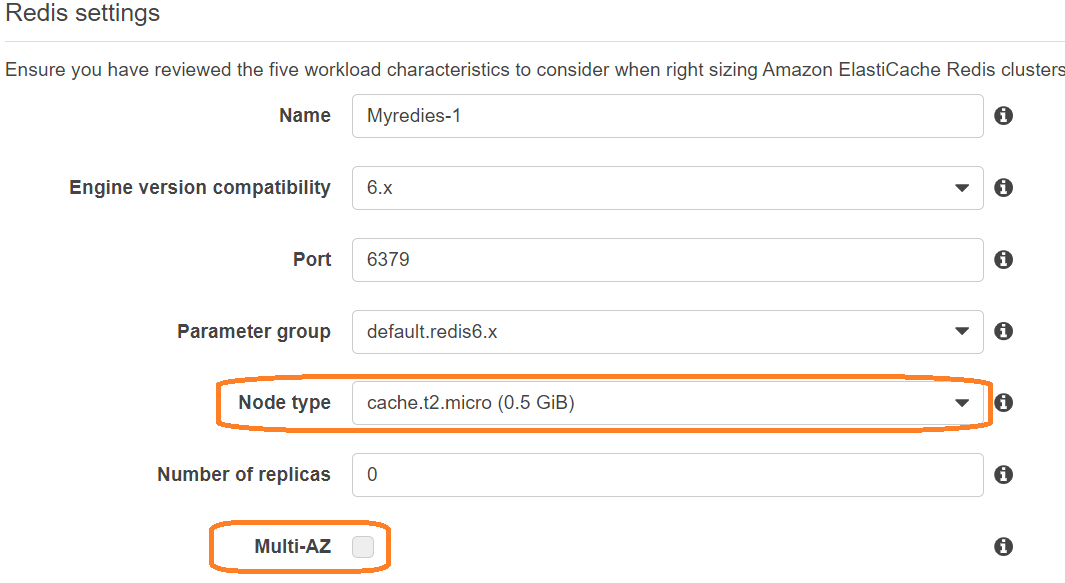
ElastiCache Dash board

**Services 🡪 Database 🡪 ElastiCache**

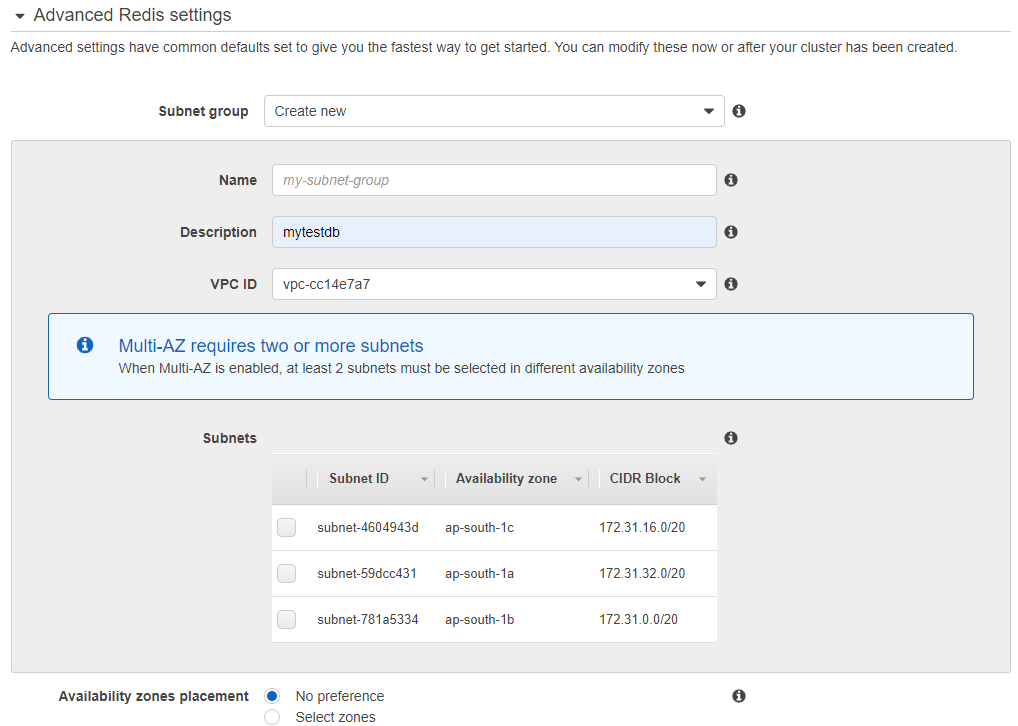
Choose **Get Started** Now button for creating ElastiCache



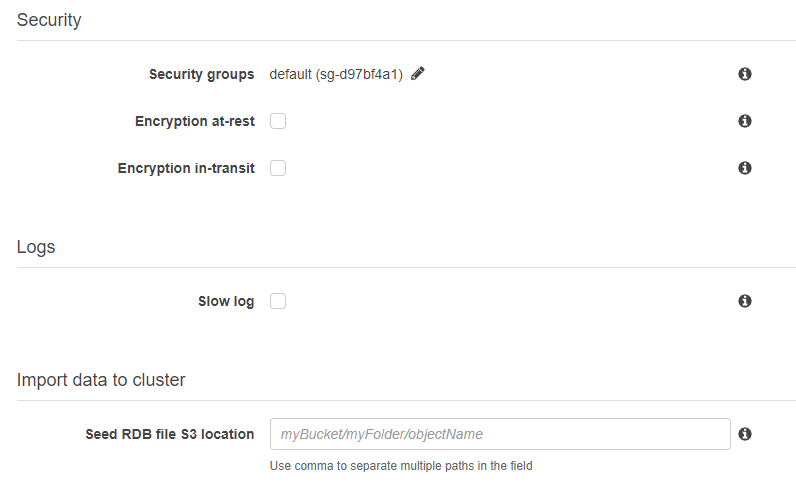


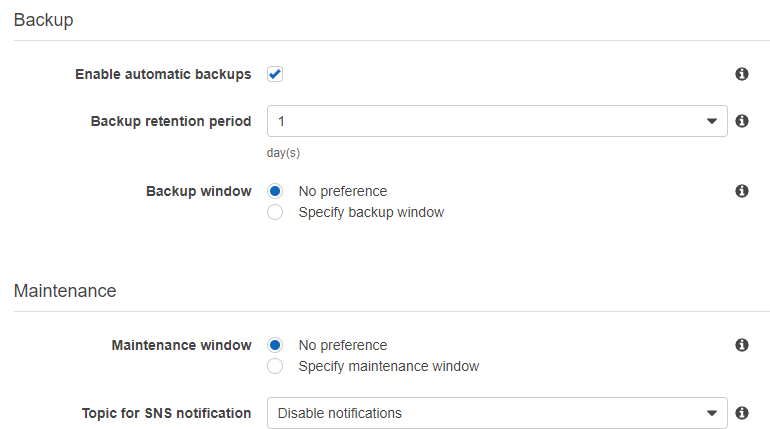


**ElastiCache Multi-AZ** provides enhanced high availability through automatic failover to a read replica, cross AZ’s, in case of a primary node failover.



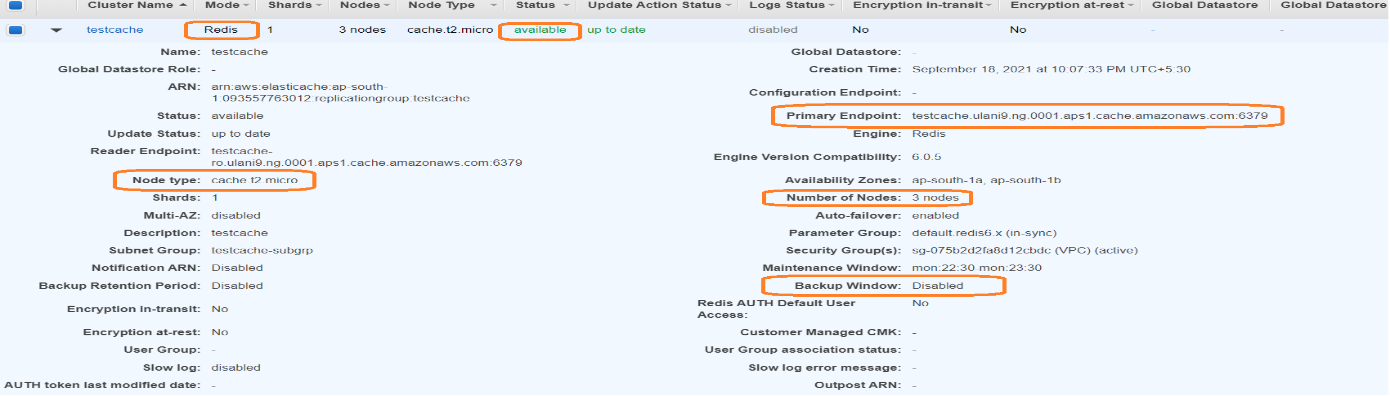
**Name** : A meaning full name that unique identifies the subnet group





**Enable automatic backups**: When sleeted, allows ElastiCache to automatically create a daily backup of a replica node

**Created ElastiCache using redis**



**Installation steps for login redis using EC2 instance**.

* yum install gcc
* yum install wget
* wget http://download.redis.io/redis-stable.tar.gz
* tar xvzf redis-stable.tar.gz
* cd redis-stable
* make /\* ubuntu system only \*/
* make /\* Amazon system \*/

--**Lookup primary endpoint of the Redis cluster in the AWS Console**

--src/redis-cli -c -h <<Primary-End-Point>>

**src/redis-cli -c -h testcache.ulani9.ng.0001.aps1.cache.amazonaws.com -p 6379**

**EC2 Console login screen:**

