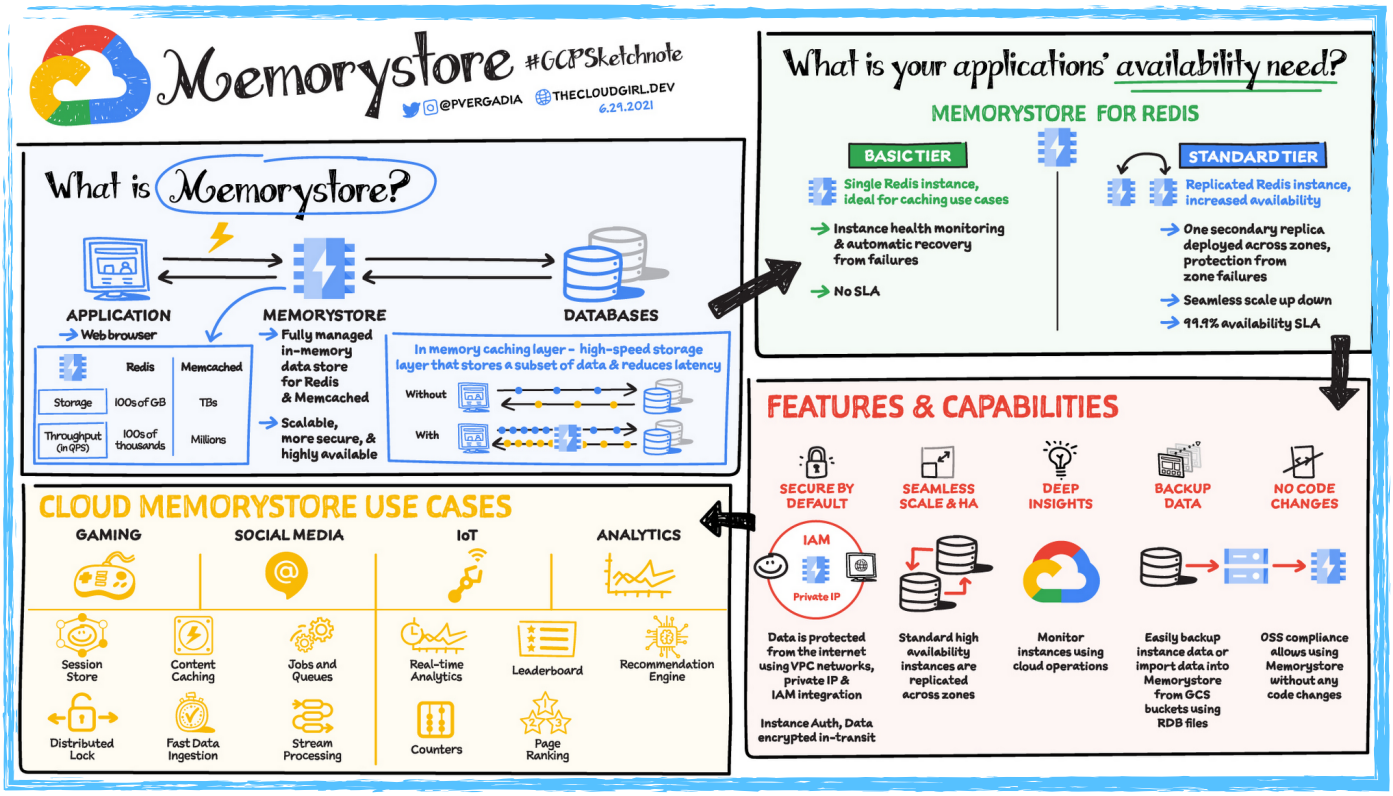
Cloud Memory store

- Memorystore is a fully managed in-memory data store service for Redis and Memcached at Google Cloud.
- It is fast, scalable, highly available, and secure. It automates complex tasks of provisioning, replication, failover, and patching so you can spend more time on other activities.
- It comes with a 99.9% SLA and integrates seamlessly with other apps in Google Cloud.
- Only Internal IP is created when you create instance from memory store.
- Memorystore is used for different types of in-memory caches and transient stores.
- This serves multiple use cases including web content caches, session stores, distributed locks, stream processing, recommendations, capacity caches, gaming leaderboards, fraud\threat detection, personalization, and ad tech.



- Two engines supported are supported in Google Cloud
    - Redis
    - Memcached
- It's a fully managed memorystore instance and sub-millisecond data access.
    - Provisioning
    - Replication
    - Failovers are fully automated.
- Import/Export data from Cloud Storage to memory store.You can export data to an RDB backup. The RDB file is written out to Cloud Storage.
- You have to enable API for Memorystore and Redis instance separately.
    - Cloud Memorystore for Memcached API
    - Cloud Memorystore for Redis API

Memorystore for Redis

- Memorystore for Redis is a fully managed Redis service for Google Cloud.
- Redis is an open-source, key-value, NoSQL database. It is an in-memory data structure that stores all the data served from memory and uses disk for storage.
- It offers a unique data model and high performance that supports various data structures like string, list, sets, hash, which it uses as a database cache or message broker. It is also called **Data Structure Server**.
- It does not support schema RDBMS, SQL or ACID transactions.
- Applications running on Google Cloud can achieve extreme performance by leveraging the highly scalable, available, secure Redis service without the burden of managing complex Redis deployments.

- Memorystore for Redis offers Basic and Standard Tiers.
    - **Basic tier**
        - The basic service tier is a zonal resource that can get you up and running for very little outlay.
        - Need to make sure that application can withstand a cold restart or full data flush
    - **Standard tier**
        - The standard tier adds cross-zone replication and automatic failover.
- We can choose how much RAM to assign to our instance up to 300 gigabytes.
    - The amount of RAM assigned to the instance will determine its maximum network throughput.
- Redis is typically used as a persistent session cache maybe to store logins or shopping carts, as a message queue to help loosely-coupled microservices, or even as a full-blown pub/sub message queue.

Features of Memorystore for Redis

The following describes the features that Memorystore for Redis provides

- **Fully managed**
    - Deploying and maintaining a Redis instance can be time consuming.
    - Memorystore for Redis provides patching, 24x7 threat monitoring, failure detection, and automatic failover, allowing you to spend more time on building your applications.

- **Simple deployment**
    - Easily deploy a single instance or highly available Redis instance by choosing Basic Tier or Standard Tier from the Google Cloud console or by automating deployment using Google Cloud CLI or Cloud Client Libraries.

- **High availability**
  - Memorystore for Redis instances in the Standard Tier are replicated across zones, monitored for health and have fast automatic failover. Standard Tier instances also provide an SLA of 99.9%.
- **Enterprise-grade security**
  - Redis instances are protected from the internet using private IPs, and access to instances is controlled and limited to applications running on the same authorized Virtual Private Cloud as the Redis instance.
  - Instances are further secured using IAM roles, which enables granular control over who can manage and access the Redis instance.
- **Scale**
  - Memorystore for Redis enables scaling your instances up to a maximum of 300 GB and supports up to 16 Gbps of network throughput.
  - With the ability to scale instance size seamlessly, you can start small and increase the size of the instance as needed.
- **Monitoring**
  - Redis metrics are available through Monitoring, making it easy to monitor your Redis instances.
  - Using Cloud Logging, you can also see the Redis logs for your instance.
- **On-demand billing**
  - With Memorystore for Redis instances, you are billed by the hour for the capacity (GB) that you provision.
  - If your instances are used for a few minutes, you are billed for only those minutes.

- Redis versions 6.x, 5.0, 4.0, and 3.2
    - The versions are always kept up to date with the latest critical patches, ensuring your Redis instances are secure.

Pricing of Memorystore for Redis

- **Service tier -** Determines whether the Redis instance is standalone or highly available.
- **Provisioned capacity -** Determines how much storage and throughput is available to the Redis instance.
- **Region -** Determines the location where the Redis instance is provisioned.
- **Replicas -** Determines how many nodes are there in a Standard Tier instance. Only applicable for instances with read replicas enabled.

Memorystore for Memcached

- Memorystore for Memcached is a fully managed Memcached service for Google Cloud.
- Memorystore for Memcached instances are provisioned **on a node basis** with vCPU and memory per cores per node, which means you can select them based on your specific application requirements
- Memcached is a simple, open-source, in-memory caching system that can be used as a temporary in-memory data storage.
- The service is built on open source Memcached and is binary and ASCII protocol compliant. This service can be accessed using standard OSS Memcached client libraries across all languages, making it easy to lift and shift existing applications with little to no code changes.
- The stored data in memory has high read and write performance and distributes data into multiple servers.
- It is a key-value of string object that is stored in memory and the API is available for all the languages.

- Applications running on Google Cloud can achieve extreme performance by leveraging the highly scalable, available, secure Memcached service without the burden of managing complex Memcached deployments.
- Memorystore also makes it very easy to deploy a Memcached service. You deploy Memorystore and Google manages the service for you, so that you can focus on your application.
- Memcached is very efficient for websites.
- Some of the common Memcached use cases include caching of reference data, database query caching, and, in some cases, use as a session store.

Features of Memorystore for Memcached

Features and Capabilities	Description
Memcached version	The service currently supports version 1.5.16
Instance sizing	• An instance can have a maximum of 20 nodes. • All nodes have the same configuration. • A node can have a minimum of 1 vCPU and a maximum of 32 vCPUs. • The minimum memory per node is 1 GB and the maximum memory supported per node is 256 GB. • Memory can be specified in 1 GB increments. The maximum size of an instance is 5 TB.
Instance scaling	• You can horizontally scale your instance by increasing or decreasing the number of nodes. • Vertically scaling your nodes requires recreating the instance.
Memcached configurations	• You can configure the instance for your specific workload.
Security and access control	• Access to an instance is restricted to clients connected to your project's VPC Network. • If the instance is using a Shared VPC network all clients in the Shared VPC network can access the instance. • Instance level authentication is not available. • Admin operations are controlled using the Identity and Access Management roles listed on the Access control page.

Platform support	Memorystore for Memcached can be accessed from the following platforms. - Compute Engine - Google Kubernetes Engine - Cloud Run (Requires Serverless VPC Access). - Cloud Functions (Requires Serverless VPC Access). - App Engine flexible and Standard environment.
Client Libraries	• All standard Memcached client libraries across all languages are supported.
Monitoring	• Memorystore for Memcached provides metrics and dashboards in the Google Cloud console to monitor the instance.

Pricing of Memorystore for Memcached

Memorystore for Memcached pricing depends on the region where the Memcached instance is provisioned and is composed of the following charges:

- **vCPUs / cores - per node**.
    - When provisioning a Memorystore for Memcached instance, you select the number of vCPUs (cores) and memory (GB) needed for a node, and specify the number of nodes you want in the instance.
- **Memory per node -** The amount of memory provisioned for each node in your cluster.
    - The total hourly cost of an instance is calculated based on the total number of vCPUs and total memory provisioned for an instance.
    - The total number of vCPUs and total memory of an instance is equal to the vCPUs and Memory for one node multiplied by the number of nodes in the instance.

- **Network pricing**
  - Memorystore for Memcached traffic must stay within the region where the instance is located.
  - There is no charge for ingress to or egress from Memorystore for Memcached.
  - Network egress charges associated with other Google Cloud services may apply when moving data into Memorystore.

Difference between Redis and Memcached

Parameter	REDIS	MEMCACHED
Cores Used	It uses single cores	It uses multiple cores
Length Of Key	In Redis, maximum key length is 2GB.	In Memcached, maximum key length is 250 bytes.
Data Structure	It uses list, string, hashes, sorted sets and bitmaps as data structure.	It uses only string and integers as data structure.
Speed	It reads and writes speed is slower than Memcached	It reads and writes speed is higher than Redis.
Replication	It supports Master-Slave Replication and Multi-Master Replication methods.	It does not support any replication method.
Durability	It is more durable than Memcached.	It is less durable than Redis.
Secondary database model	It has Document Store, Graph DBMS, Search Engine, and Time Series DBMS as secondary database models.	It has no secondary database models.
Persistence	It uses persistent data.	It does not use persistent data.
Partitioning method	It supports Sharding.	It does not support any partitioning method.
Initial Release	It was released in 2009.	It was released in 2003.
Developer	It was developed by Salvatore Sanfilippo.	It was developed by Danga Interactive.