Section 16: planning and configuring storage

Storage overview

- → object storage
  - ↳ cloud storage
- → Instance Storage
  - ↳ persistent Disks
- → Database Storage
  - → SQL
    - ↳ cloud SQL
    - ↳ Cloud Spanner
  - → No SQL
    - ↳ Cloud Data / File Store → Docs
    - ↳ Bigtable → columnar
  - → Analytic
    - ↳ Big Query

Cloud Storage

- → unstructured Data
  - ↳ Images + video + Text
- → Archived Data
- → Temporary Storage [moving data]

Local organization

→ Global Name Space

↓

Buckets

↓

Folders

Classes

- → Standard Storage
  - * frequent access
  - * performance optimized
- → Archive + Backup storage
  - ↳ Nearline
    - ↳ stored at least 30 days
  - ↳ Cold line
    - ↳ at least 90 days
  - ↳ Archive
    - ↳ at least 365 days

* there is also a read/write guidance once per x days
x = 30/90/365

Redundancy

- → Regional
  - ↳ Single Region
  - ↳ Replicated across zones
- → Dual-Region
  - ↳ 2 Regions
  - ↳ auto failover
- → Multi-Region
  - ↳ US, Europe, Asia
  - ↳ auto-failover

Object Lifecycle Policies

- → automatically transition to cheaper tier
- → Retention Policies (time)
  - ↳ Minimum period
- → Hold delete prevention

Security

- → All encrypted
  - → google keys
  - → customer cloud key
  - → customer supplied
- → Bucket acces can be managed with
  - ↳ Bucket level Rules
  - ↳ Iam
- → automatic logging

Cloud SQL

- → Hardware options
- → auto backups
- → High availability
  - ↳ Replication
- → MySQL Database Flags
- → Maintenance Schedule

Database Info can be found in instance details

Cloud Spanner

- → Scalable Horizontally
- → Strong consitency
- → availability around the world

Together with SQL Relational db options

Pick Multi-zone deployment

Spanner is expensive!

Cloud Data store

NoSQL Database

Modes
- → Native Mode
- → DataStore Mode

Serverless Service

One Database per project

DataStore is created automatically
- ↳ Api Based
- ↳ entities cost Money

Entity

properties can be different per Row because of noSQL

Cloud Firestore

- → Managed NOSQL
- → Document Based
  - ↳ key-value
  - ↳ Hierarchial
  - ↳ Json like

Use when
- ↳ Semi-structured Data
  - ↳ Not Fixed Schema
  - ↳ Different attributes across schema
- → Query on Multiple attribute
- → Ingestion volume not too high

2 Modes
- → DataStore Mode
  - * Backend for server apps
  - * No syncing with Mobile device
- → Native Mode
  - * Mobile + Web apps
  - * Large Number of connections

Cloud Bigtable

- → No SQL
- → Wide-column
- → low-latency + high availability
- → works with analytics
- → Not serverless

Bigtable is expensive!

Type

- → Production
- → Development

Hardware options

Node based

↳ Metadata is stored in nodes telling services where data is located

For:
- * High volume
- * High velocity

Based on collossus

Cloud Memorystore

↳ Cacheing Data
- ↳ DB's
- ↳ Games
- ↳ Data streams

→ Redis

- * connect from lots of different services
- * up to 300 gb
- * 12 GPS throughput

Redis Protocol Compliance Service

- → Basic Tier
  - → no replication
  - → no failover
- → Standard tier
  - → Replication
  - → fail over

→ MemCached

In-Memory key-value store
- ↳ Distributed

For:

Reference data
Query cache & cache Results
Session cache

- → Is one cluster
  - ↳ Max 20 Nodes
  - ↳ Max 5 Tb memory
    - ↳ so good for large cache Needs
- → supports different services.