Google Cloud

Partner Certification Academy

Associate Cloud Engineer

The information in this presentation is classified:

Google confidential & proprietary

⚠ This presentation is shared with you under NDA.

- Do **not** record or take screenshots of this presentation.
- Do **not** share or otherwise distribute the information in this presentation with anyone **inside** or **outside** of your organization.

Thank you!

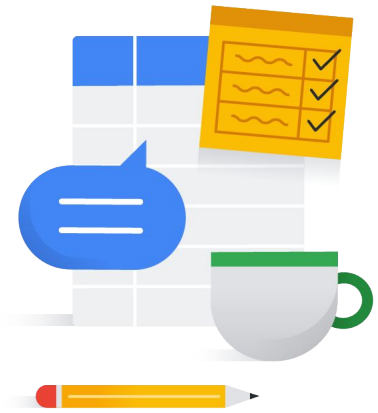
Google Cloud

Session logistics

- When you have a question, please:
  - Click the Raise hand button in Google Meet.
  - Or add your question to the Q&A section of Google Meet.
  - Please note that answers may be deferred until the end of the session.
- These slides are available in the Student Lecture section of your Qwiklabs classroom.
- The session is **not recorded.**
- Google Meet does not have persistent chat.
  - If you get disconnected, you will lose the chat history.
  - Please copy any important URLs to a local text file as they appear in the chat.

Program issues or concerns?

- Problems with **accessing** Cloud Skills Boost for Partners
  - partner-training@google.com
- Problems with **a lab** (locked out, etc.)
  - support@qwiklabs.com
- Problems with accessing Partner Advantage
  - https://support.google.com/googlecloud/topic/9198654



Associate Cloud Engineer

The Google Cloud Certified

Associate Cloud Engineer exam assesses your ability to:

- Setup a cloud solution environment
- Plan and configure a cloud solution
- Deploy and implement a cloud solution
- Ensure successful operation of a cloud solution
- Configure access and security

For more information:

https://cloud.google.com/certification/cloud-engineer

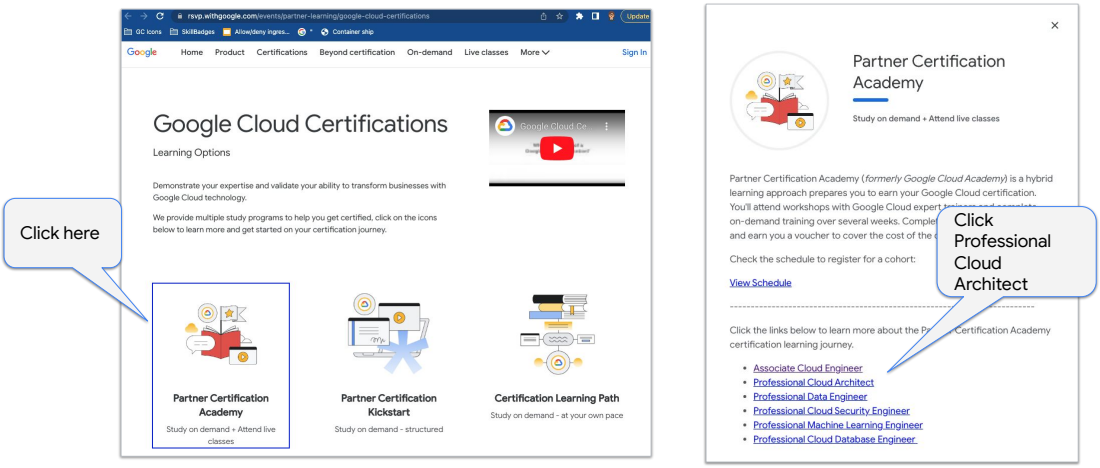
Associate Cloud Engineer
https://cloud.google.com/certification/cloud-engineer

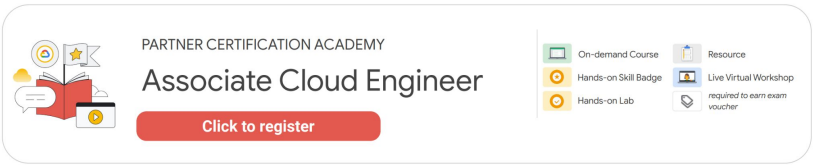
Exam Guide
https://cloud.google.com/certification/guides/cloud-engineer

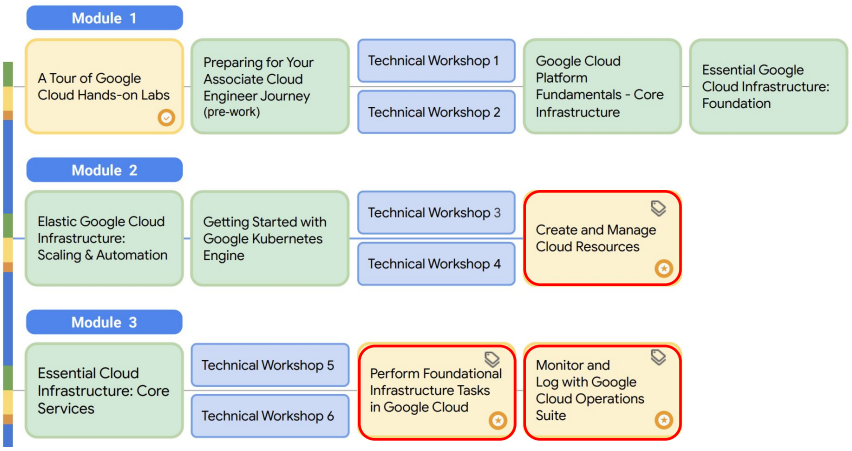
Sample Questions
https://docs.google.com/forms/d/e/1FAIpQLSfexWKtXT2OSFJ-obA4iT3GmzgiOCGvjrT9OfxilWC1yPtmfQ/viewform

Learning Path - Partner Certification Academy Website

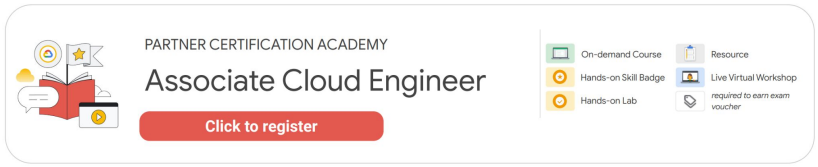
Go to: https://rsvp.withgoogle.com/events/partner-learning/google-cloud-certifications

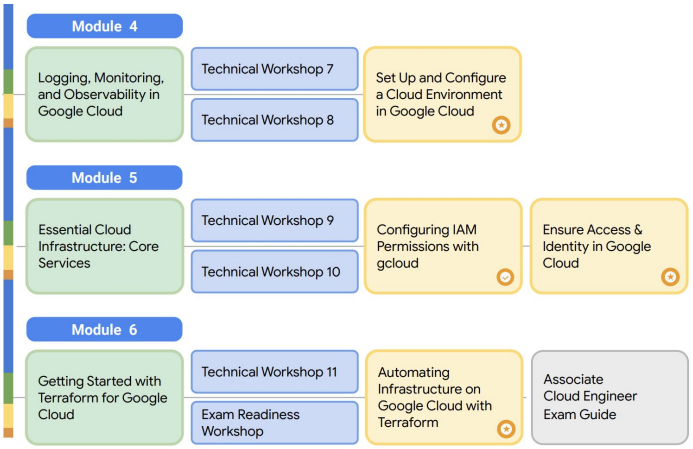






Needed for Exam Voucher





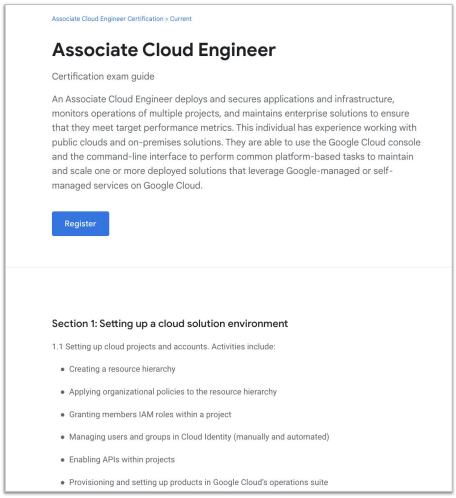
Associate Cloud Engineer (ACE) Exam Guide

Each module of this course covers Google Cloud services based on the topics in the ACE Exam Guide

The primary topics are:

- Compute Engine
- VPC Networks
- Google Kubernetes Engine
- Cloud Run, Cloud Functions and App Engine
- Storage and database options
- Resource Hierarchy/Identity and Access Management (IAM)
- Logging and Monitoring

Next discussion



Associate Cloud Engineer Certification > Current

Associate Cloud Engineer

Certification exam guide

An Associate Cloud Engineer deploys and secures applications and infrastructure, monitors operations of multiple projects, and maintains enterprise solutions to ensure that they meet target performance metrics. This individual has experience working with public clouds and on-premises solutions. They are able to use the Google Cloud console and the command-line interface to perform common platform-based tasks to maintain and scale one or more deployed solutions that leverage Google-managed or self-managed services on Google Cloud.

Register

Section 1: Setting up a cloud solution environment

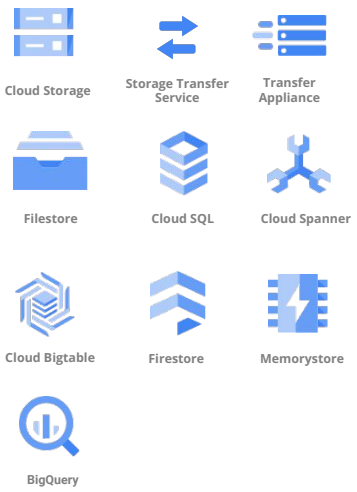
1.1 Setting up cloud projects and accounts. Activities include:

- Creating a resource hierarchy
- Applying organizational policies to the resource hierarchy
- Granting members IAM roles within a project
- Managing users and groups in Cloud Identity (manually and automated)
- Enabling APIs within projects
- Provisioning and setting up products in Google Cloud's operations suite

https://cloud.google.com/certification/guides/cloud-engineer/

Storage and database options

Exam Guide Overview - Storage and Data Transfer Options



2.3 Planning and configuring data storage options, including:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore, Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

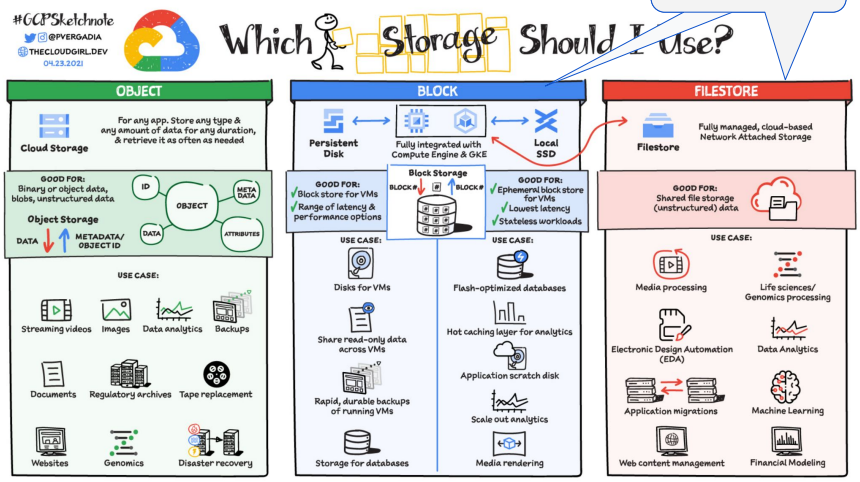
4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore, Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Storage Overview

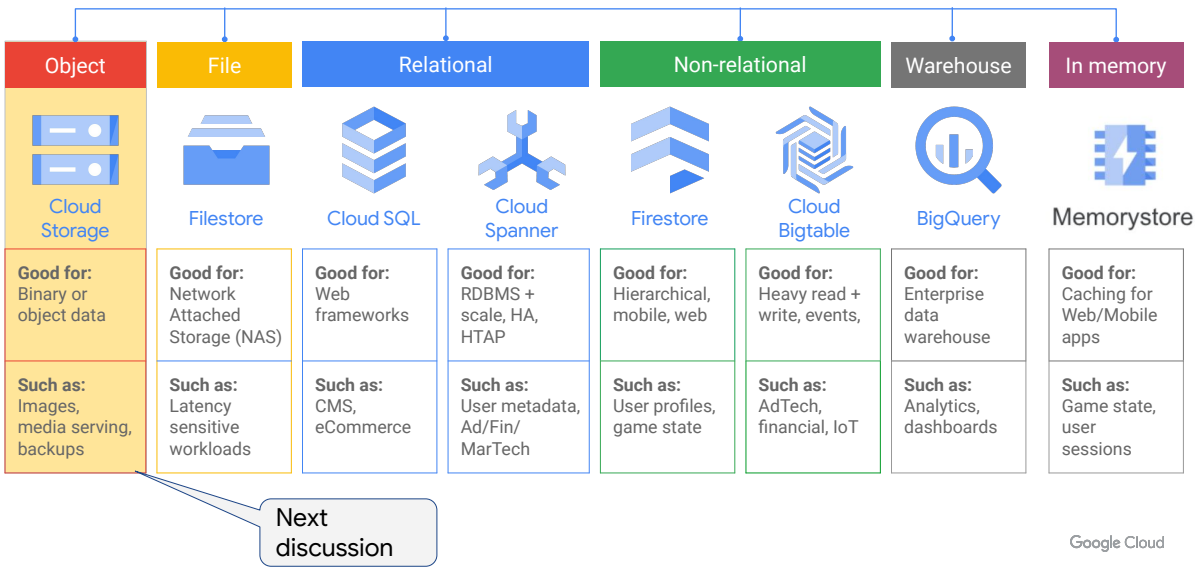
Compute Engine topics

Object storage is covered next



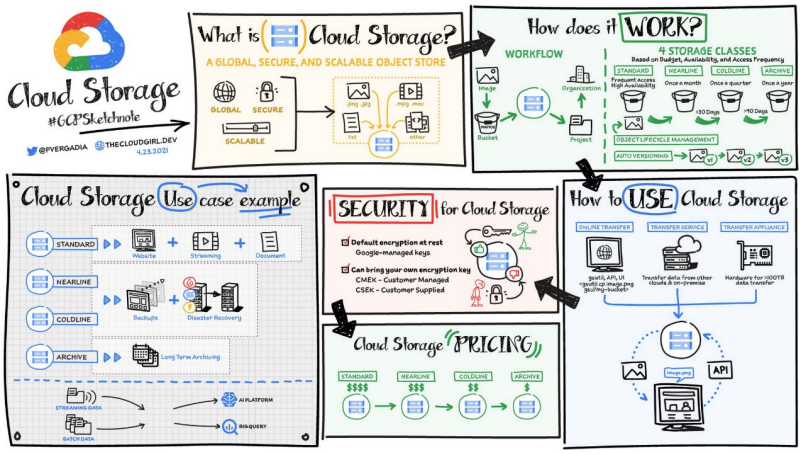
A map of storage options in Google Cloud

Storage and database services



This table shows the storage and database services and highlights the storage service type (object, file, relational, non-relational, and data warehouse), what each service is good for, and intended use.

All you need to know about Cloud Storage

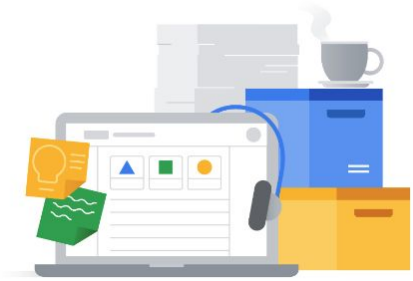


https://cloud.google.com/blog/topics/developers-practitioners/all-you-need-know-about-cloud-storage

Cloud Storage is a fully managed storage service

Binary large-object (BLOB) storage used for

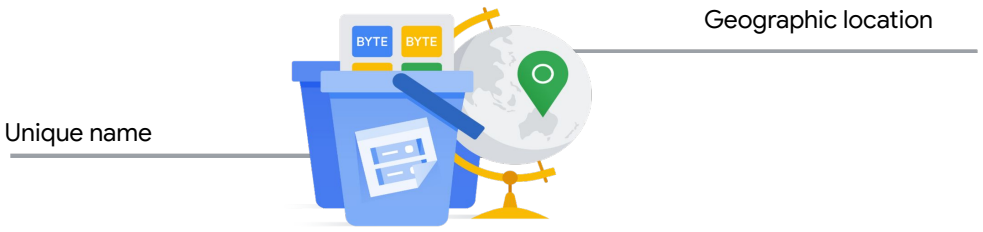
- Online content
- Backup and archiving
- Storage of intermediate results
- And much more....



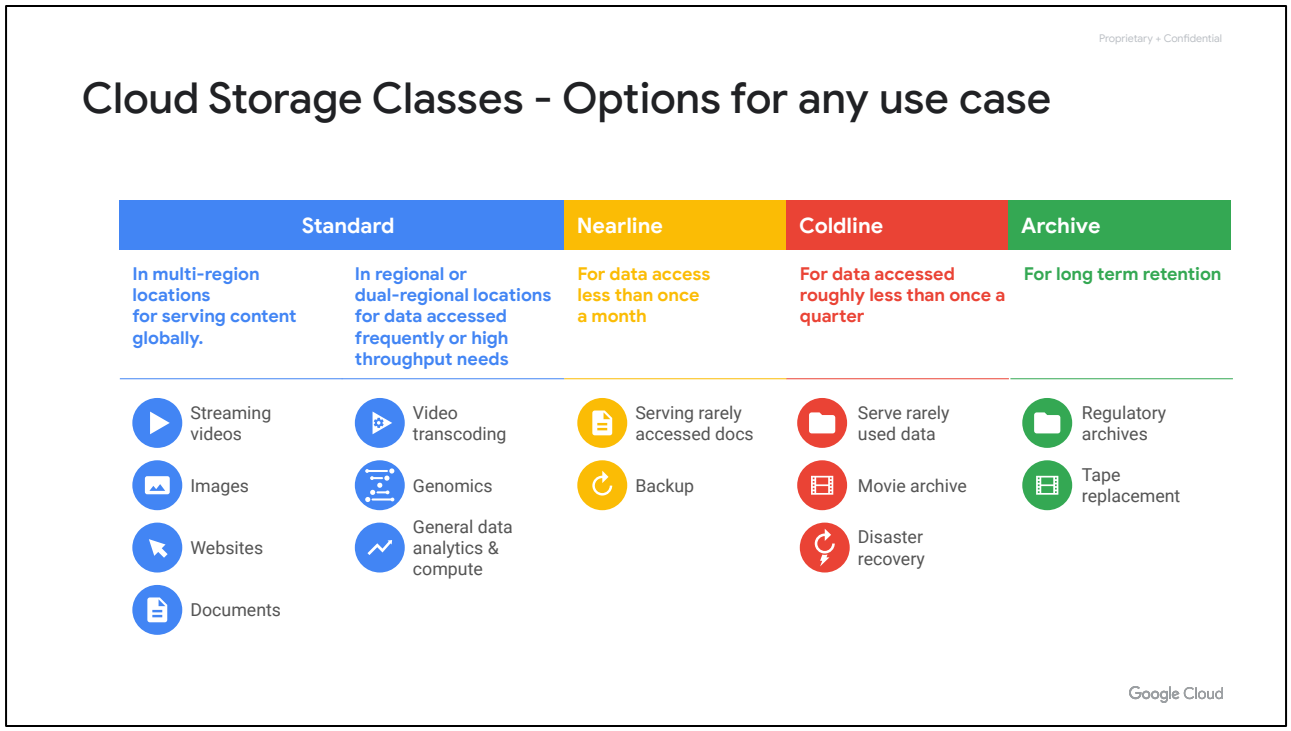
Object storage for companies of all sizes
https://cloud.google.com/storage

Cloud Storage's primary use is whenever binary large-object storage (also known as a "BLOB") is needed for online content such as videos and photos, for backup and archived data, and for storage of intermediate results in processing workflows.

Files are organized into buckets



Cloud Storage files are organized into buckets. A bucket needs a globally-unique name and a specific geographic location for where it should be stored, and an ideal location for a bucket is where latency is minimized. For example, if most of your users are in Europe, you probably want to pick a European location so either a specific Google Cloud region in Europe, or else the EU multi-region.



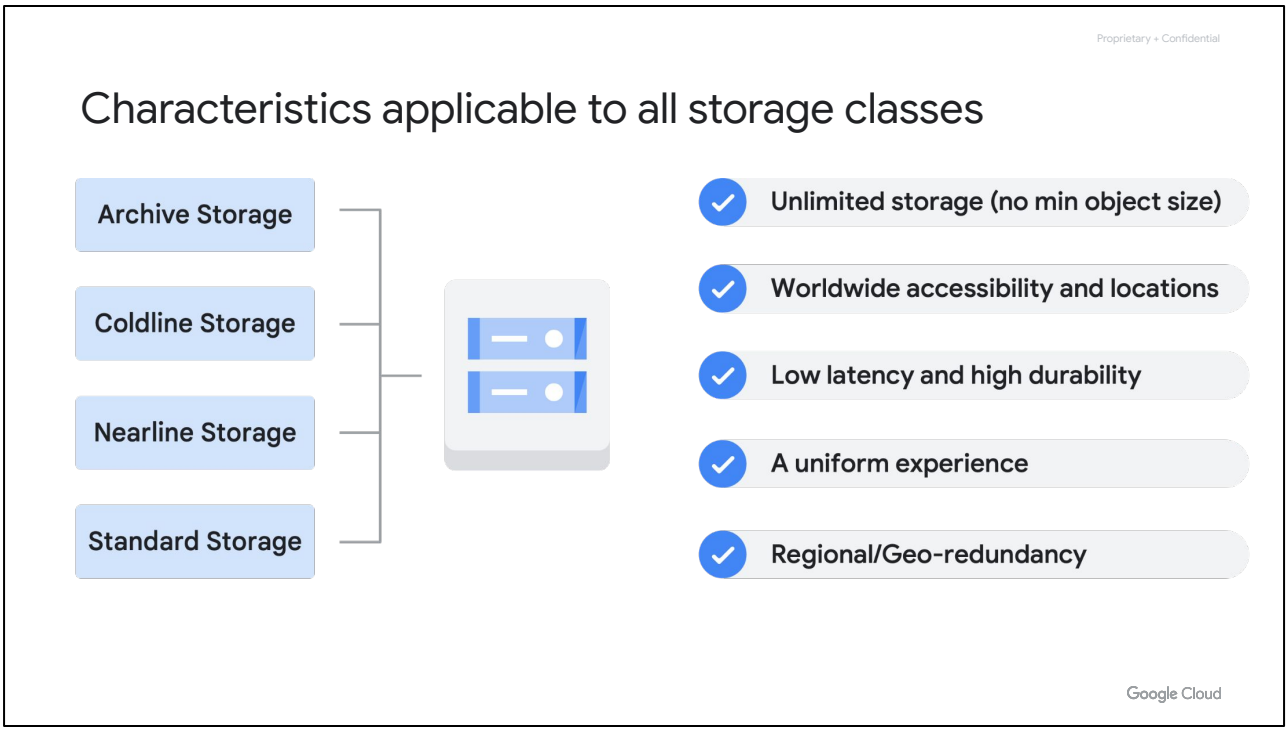
There are four primary storage classes in Cloud Storage and stored data is managed and billed according to which “class” it belongs.

The first is Standard Storage. Standard Storage is considered best for frequently accessed, or “hot,” data. It’s also great for data that is stored for only brief periods of time.

The second storage class is Nearline Storage. This is best for storing infrequently accessed data, like reading or modifying data once per month or less, on average. Examples might include data backups, long-tail multimedia content, or data archiving.

The third storage class is Coldline Storage. This is also a low-cost option for storing infrequently accessed data. However, as compared to Nearline Storage, Coldline Storage is meant for reading or modifying data, at most, once every 90 days.

The fourth storage class is Archive Storage. This is the lowest-cost option, used ideally for data archiving, online backup, and disaster recovery. It’s the best choice for data that you plan to access less than once a year, because it has higher costs for data access and operations and a 365-day minimum storage duration.

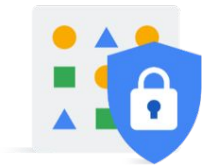


Although each of these four classes has differences, it's worth noting that several characteristics apply across all these storage classes.
These include:

- Unlimited storage with no minimum object size requirement,
- Worldwide accessibility and locations,
- Low latency and high durability,
- A uniform experience, which extends to security, tools, and APIs, and
- Geo-redundancy if data is stored in a multi-region or dual-region. So this means placing physical servers in geographically diverse data centers to protect against catastrophic events and natural disasters, and load-balancing traffic for optimal performance.

Additional Cloud Storage features

- Pay only for what you use
- No prior provisioning of capacity
- Encrypts data on the server side
- Use HTTPS/TLS (Transport Layer Security)



Cloud Storage has no minimum fee because you pay only for what you use, and prior provisioning of capacity isn't necessary.

And from a security perspective, Cloud Storage always encrypts data on the server side, before it's written to disk, at no additional charge. Data traveling between a customer's device and Google is encrypted by default using HTTPS/TLS (Transport Layer Security).

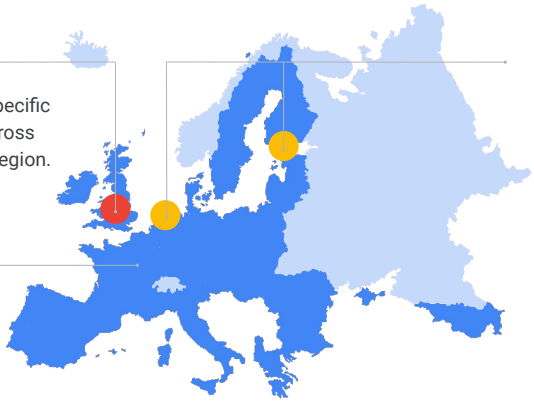
Choosing a location type

Regional

Your data is stored in a specific region with replication across availability zones in that region.

Multi-regional

Your data is distributed redundantly across US, EU or Asia.



Dual-regional

Your data is replicated across a specific pair of regions.

How to choose between regional, dual-region and multi-region Cloud Storage

Google Cloud

Bucket locations
https://cloud.google.com/storage/docs/locations

How to migrate Cloud Storage data from multi-region to regional
https://cloud.google.com/blog/products/storage-data-transfer/multi-region-google-cloud-storage-to-regional-data-migration

How to choose between regional, dual-region and multi-region Cloud Storage
https://cloud.google.com/blog/products/storage-data-transfer/choose-between-regional-dual-region-and-multi-region-cloud-storage/

Choosing a storage classes

	Standard	Nearline	Coldline	Archive
Use case	"Hot" data and/or stored for only brief periods of time like data-intensive computations	Infrequently accessed data like data backup, long-tail multimedia content, and data archiving	Infrequently accessed data that you read or modify at most once a quarter	Data archiving, online backup, and disaster recovery
Minimum storage duration*	None	30 days	90 days	365 days
Retrieval cost	None	$0.01 per GB	$0.02 per GB	$0.05 per GB
Availability SLA	99.95% (multi/dual) 99.90% (region)	99.90% (multi/dual) 99.00% (region)		None
Durability	99.999999999%			

*Minimum storage duration = if delete file before x days, will still pay for x days

Cloud Storage has four storage classes: Standard, Nearline, Coldline and Archive and each of those storage classes provide 3 location types:

- A multi-region is a large geographic area, such as the United States, that contains two or more geographic places.
- A dual-region is a specific pair of regions, such as Finland and the Netherlands.
- A region is a specific geographic place, such as London.

Objects stored in a multi-region or dual-region are geo-redundant.

- **Standard** Storage is best for data that is frequently accessed ("hot" data) and/or stored for only brief periods of time. This is the most expensive storage class but it has no minimum storage duration and no retrieval cost. When used in a:
  - region, Standard Storage is appropriate for storing data in the same location as Google Kubernetes Engine clusters or Compute Engine instances that use the data. Co-locating your resources maximizes the performance for data-intensive computations and can reduce network charges.
  - dual-region, you still get optimized performance when accessing Google Cloud products that are located in one of the associated regions, but you also get the improved availability that comes from

    - storing data in geographically separate locations.
    - multi-region, Standard Storage is appropriate for storing data that is accessed around the world, such as serving website content, streaming videos, executing interactive workloads, or serving data supporting mobile and gaming applications.
- **Nearline** Storage is a low-cost, highly durable storage service for storing infrequently accessed data like data backup, long-tail multimedia content, and data archiving. Nearline Storage is a better choice than Standard Storage in scenarios where slightly lower availability, a 30-day minimum storage duration, and costs for data access are acceptable trade-offs for lowered at-rest storage costs.
- **Coldline** Storage is a very-low-cost, highly durable storage service for storing infrequently accessed data. Coldline Storage is a better choice than Standard Storage or Nearline Storage in scenarios where slightly lower availability, a 90-day minimum storage duration, and higher costs for data access are acceptable trade-offs for lowered at-rest storage costs.
- **Archive** Storage is the lowest-cost, highly durable storage service for data archiving, online backup, and disaster recovery. Unlike the "coldest" storage services offered by other Cloud providers, your data is available within milliseconds, not hours or days. Unlike other Cloud Storage storage classes, Archive Storage has no availability SLA, though the typical availability is comparable to Nearline Storage and Coldline Storage. Archive Storage also has higher costs for data access and operations, as well as a 365-day minimum storage duration. Archive Storage is the best choice for data that you plan to access less than once a year.

Let’s focus on durability and availability. All of these storage classes have 11 nines of durability, but what does that mean? Does that mean you have access to your files at all times? No, what that means is you won't lose data. You may not be able to access the data which is like going to your bank and saying well my money is in there, it's 11 nines durable. But when the bank is closed we don't have access to it which is the availability that differs between the storage classes and the location type.

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

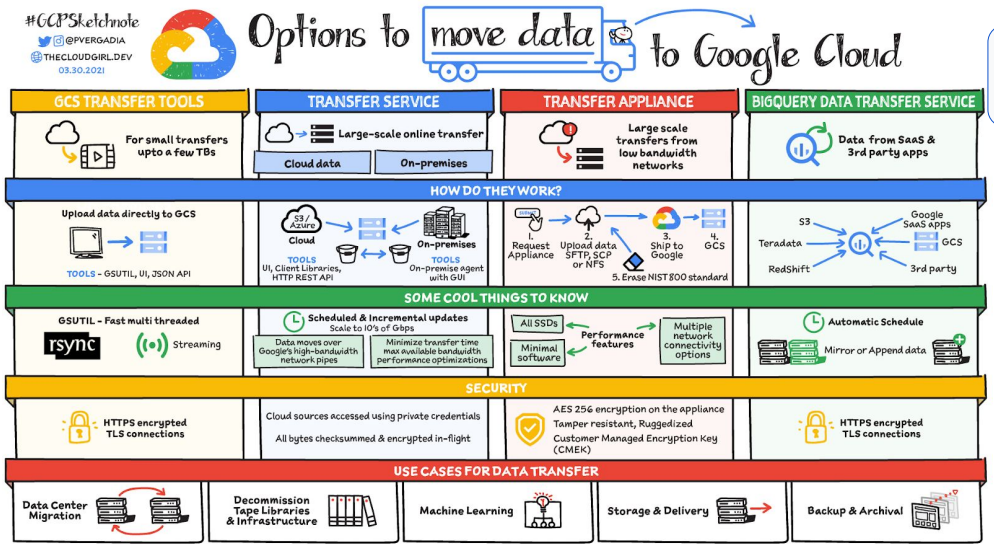
- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore, Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery



BigQuery discussed later

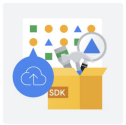
[How to transfer data to Google Cloud?]

How to transfer data to Google Cloud
https://www.youtube.com/watch?v=It9bOxlsKs4
https://cloud.google.com/blog/topics/developers-practitioners/how-transfer-your-data-google-cloud

Bringing data into Cloud Storage

- Online transfer
  - Drag and drop in the Goggle Cloud Console
  - Upload / download via the command line
    - gsutil/gcloud
- Storage Transfer Service
  - Create jobs to run once or on a scheduled bases
  - Transfer data from
    - Other clouds (AWS, Azure), URL, Posix filesystem
    - On-premise
    - Cloud Storage Bucket to Cloud Storage Bucket
- Transfer Appliance
  - Rackable, high-capacity storage server leased from Google
  - Used when the amount of data to be transferred would take too much time given network bandwidth between the customer location and Google

How long will it take to transfer data?



Online transfer


Storage Transfer Service


Transfer Appliance

Cloud Storage - data transfer options:
https://cloud.google.com/architecture/migration-to-google-cloud-transferring-your-large-datasets#transfer-options

How long will it take to transfer data?
https://cloud.google.com/architecture/migration-to-google-cloud-transferring-your-large-datasets#time

Online transfer
gsutil: https://cloud.google.com/storage/docs/gsutil
gcloud: https://cloud.google.com/sdk/gcloud/reference/storage

Storage Transfer Service
https://cloud.google.com/storage-transfer/docs/overview

Transfer Appliance
https://cloud.google.com/transfer-appliance

Transfer appliance Youtube video:
https://www.youtube.com/watch?v=4g2ntSRU2pI

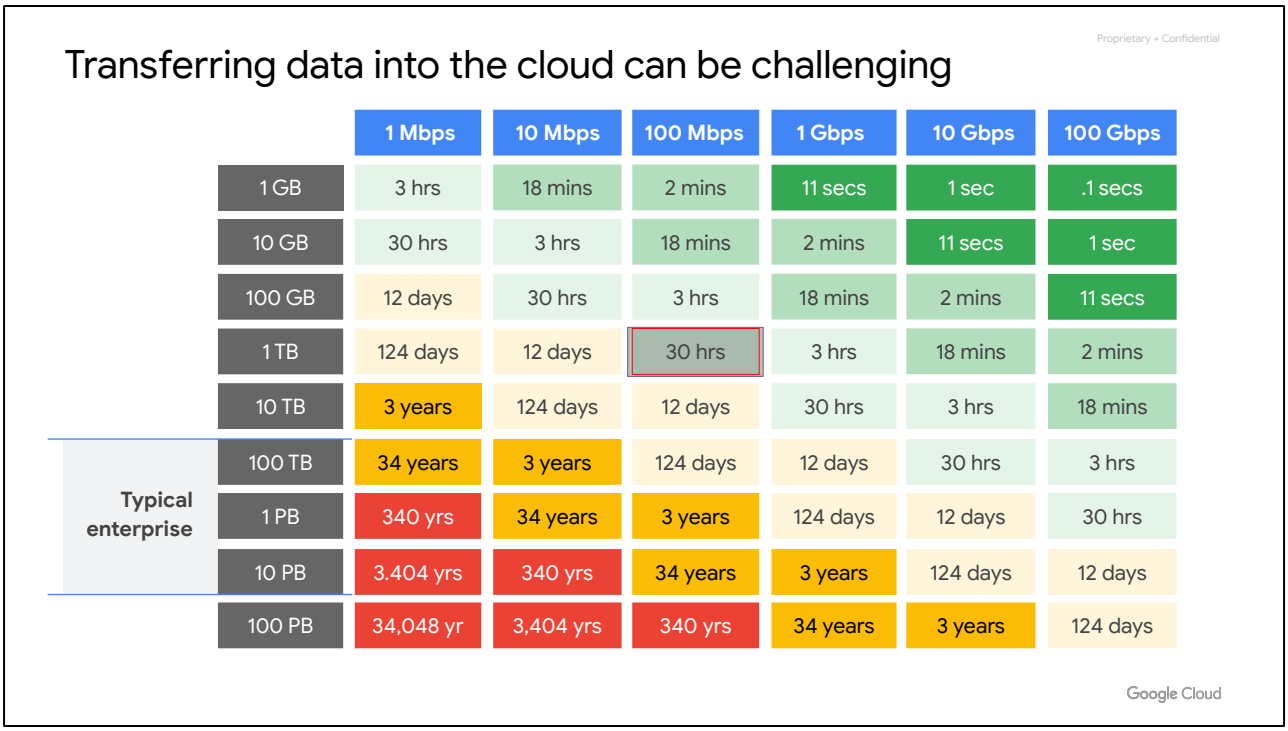
There are several ways to bring your data into Cloud Storage.

Many customers simply carry out their own online transfer using gsutil, which is the

Cloud Storage command from the Cloud SDK. Data can also be moved in by using a drag and drop option in the Google Cloud console, if accessed through the Google Chrome web browser.

But what if you have to upload terabytes or even petabytes of data? **Storage Transfer Service** enables you to import large amounts of online data into Cloud Storage quickly and cost-effectively. The Storage Transfer Service lets you schedule and manage batch transfers to Cloud Storage from another cloud provider, from a different Cloud Storage region, or from an HTTP(S) endpoint.

And then there is the **Transfer Appliance**, which is a rackable, high-capacity storage server that you lease from Google Cloud. You connect it to your network, load it with data, and then ship it to an upload facility where the data is uploaded to Cloud Storage. You can transfer up to a petabyte of data on a single appliance.



Data transfer speeds
https://cloud.google.com/transfer-appliance/docs/4.0/overview#transfer-speeds

One of the data transfer barriers that all companies face is their available network. The rate at which we are creating data is accelerating far beyond our capacity to send it all over typical networks. So all companies battle with this

In working with our customers we've found the typical enterprise organization has about 10 petabytes of information and available network bandwidth of somewhere between 100 Mbps and 1 Gbps. This works out to somewhere between 3 and 34 years of ingress. Too long. Now all of that data has different storage requirements, use cases and will likely have many different eventual homes. Because of this organizations need data transfer options across this x and y axis.

Saving files to Cloud Storage - `gsutil`

- `gsutil` - Command line to upload/download files
  - For smaller amounts of data (<1TB) if have adequate bandwidth

- Upload

```
gsutil cp OBJECT_LOCATION gs://DESTINATION_BUCKET_NAME/
gsutil cp desktop/myfile.png gs://my-bucket/
```

- Download

```
gsutil cp gs://BUCKET_NAME/OBJECT_NAME SAVE_TO_LOCATION
gsutil cp gs://my-bucket/* desktop/file-folder/
```

New: **`gcloud storage`** commands were introduced in 2022

Cloud Storage - command line upload:
https://cloud.google.com/storage/docs/uploading-objects#prereq-cli

Cloud Storage - command line download:
https://cloud.google.com/storage/docs/downloading-objects

Gsutil copy syntax:
https://cloud.google.com/storage/docs/gsutil/commands/cp#description

Creating and managing data transfers programmatically:
https://cloud.google.com/storage-transfer/docs/create-manage-transfer-program

Create and manage data transfers with gcloud (Cloud Storage):
https://cloud.google.com/storage-transfer/docs/create-manage-transfer-gcloud

Summary: Choosing a transfer option

Where you're moving data from	Scenario	Suggested products
Another cloud provider (for example, Amazon Web Services or Microsoft Azure) to Google Cloud	—	Storage Transfer Service
Cloud Storage to Cloud Storage (two different buckets)	—	Storage Transfer Service
Your private data center to Google Cloud	Enough bandwidth to meet your project deadline for less than 1 TB of data	gsutil
Your private data center to Google Cloud	Enough bandwidth to meet your project deadline for more than 1 TB of data	Storage Transfer Service for on-premises data
Your private data center to Google Cloud	Not enough bandwidth to meet your project deadline	Transfer Appliance

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

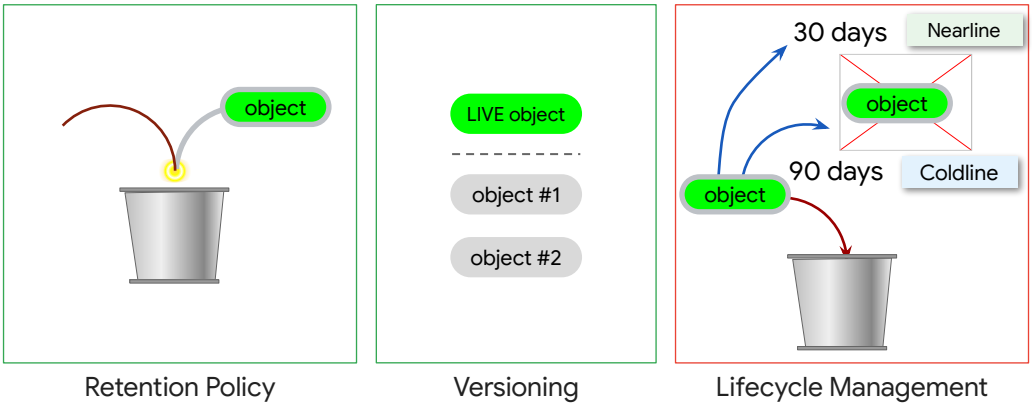
3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore, Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Cloud Storage has many object management features



Best practices for Cloud Storage cost optimization
https://cloud.google.com/blog/products/storage-data-transfer/best-practices-for-cloud-storage-cost-optimization

Cloud Storage has many object management features. For example, you can set a retention policy on all objects in the bucket. For example, the objects should be expired after 30 days.

You can also use versioning, so that multiple versions of an object are tracked and available if necessary. You might even set up lifecycle management, to automatically move objects that haven't been accessed in 30 days to Nearline and after 90 days to Coldline.

Options for controlling data lifecycles

- A **retention policy** specifies a retention period to be placed on a bucket.
  - An object cannot be deleted or replaced until it reaches the specified age.
- **Object Versioning** can be enabled on a bucket in order to retain older versions of objects.
  - When the live version of an object is deleted or replaced, it becomes noncurrent
  - If a live object version is accidentally deleted, can restore the noncurrent version back to the live version.
  - Object Versioning increases storage costs, but can be mitigated by Lifecycle Management to delete older objects
- **Object Lifecycle Management** can be configured for a bucket, which provides automated control over deleting objects and changing storage classes

Options for controlling data lifecycles
https://cloud.google.com/storage/docs/control-data-lifecycles

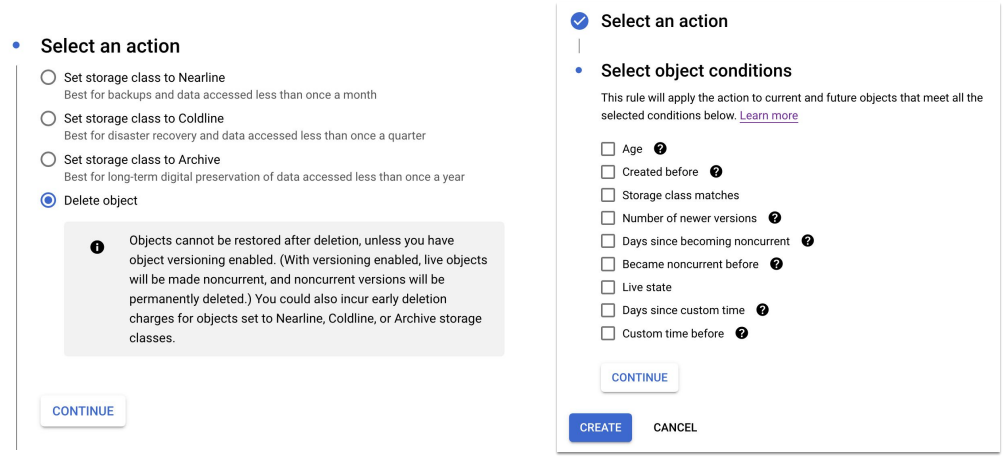
Object Lifecycle Management:
https://cloud.google.com/storage/docs/lifecycle

Lifecycle conditions:
https://cloud.google.com/storage/docs/lifecycle#conditions

gsutil command:
https://cloud.google.com/storage/docs/gsutil/commands/lifecycle

Object versioning:
https://cloud.google.com/storage/docs/object-versioning

Set Lifecycle policy - Console



Object Lifecycle Management
https://cloud.google.com/storage/docs/lifecycle

Set Lifecycle Policy - CLI

Create a JSON config file containing the rules

Then run:

```
gsutil lifecycle set
[config-file]
gs://acme-data-bucket
```

```
{
"lifecycle": {
  "rule": [
  {
    "action": {"type": "Delete"},
    "condition": {
      "numNewerVersions": 2,
      "isLive": false
    }
  },
  {
    "action": {"type": "Delete"},
    "condition": {
      "daysSinceNoncurrentTime": 7
    }}]}}
```

Rule 1: Delete noncurrent objects if there are 2 newer ones

Rule 2: Delete noncurrent versions of objects after they've been noncurrent for 7 days.

Lifecycle - Get or set lifecycle configuration for a bucket
https://cloud.google.com/storage/docs/gsutil/commands/lifecycle

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore, Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

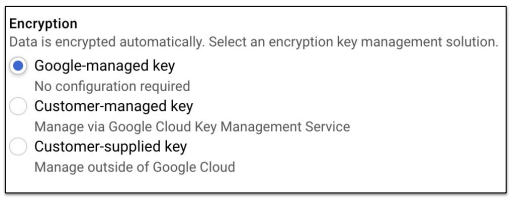
Cloud Storage encryption options

- Default - Google Cloud manages encryption
- Customer-managed encryption keys (CMEK)
  - Encryption occurs after Cloud Storage receives data but before the data is written to disk
    - Keys are managed through Cloud Key Management Service (KMS)
- Customer-supplied encryption keys (CSEK)
  - Keys created and managed externally to Google Cloud
    - Keys act as an additional encryption layer on top of the Google default encryption

---

- Client-side encryption (external to Google Cloud)
  - Data is sent to Cloud Storage already encrypted
    - Data also undergoes server-side encryption by Google

Google Cloud



Bucket access permissions (IAM) is discussed in a different module

Customer

Lab - Getting Started with Cloud KMS
https://partner.cloudskillsboost.google/catalog_lab/368

Provide support for external keys with EKM
You can encrypt data in BigQuery and Compute Engine with encryption keys that are stored and managed in a third-party key management system that's deployed outside Google's infrastructure. External Key Manager allows you to maintain separation between your data at rest and your encryption keys while still leveraging the power of cloud for compute and analytics.
https://cloud.google.com/security-key-management

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

Discussed in Compute Engine module.

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore, Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

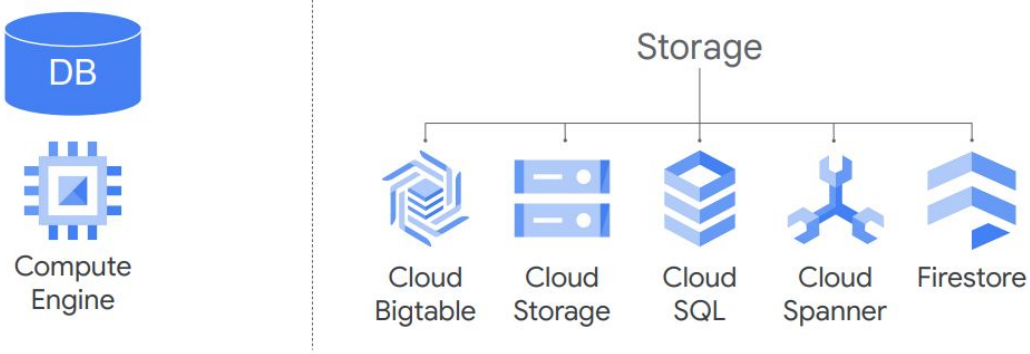
3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

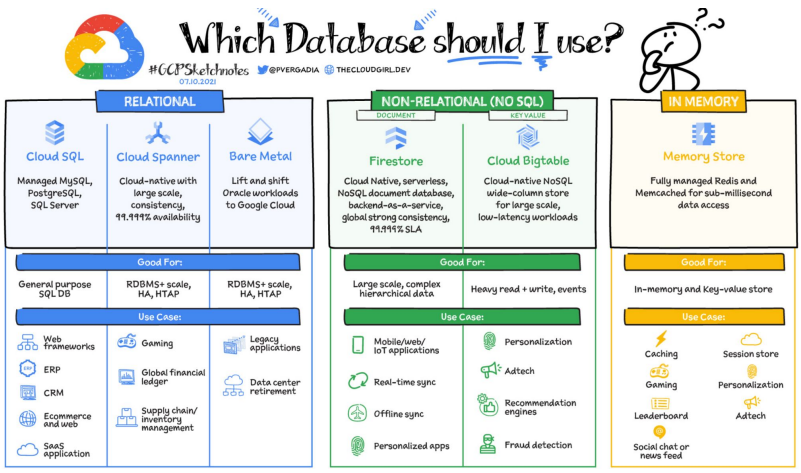
- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore , Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Custom and Managed Solutions



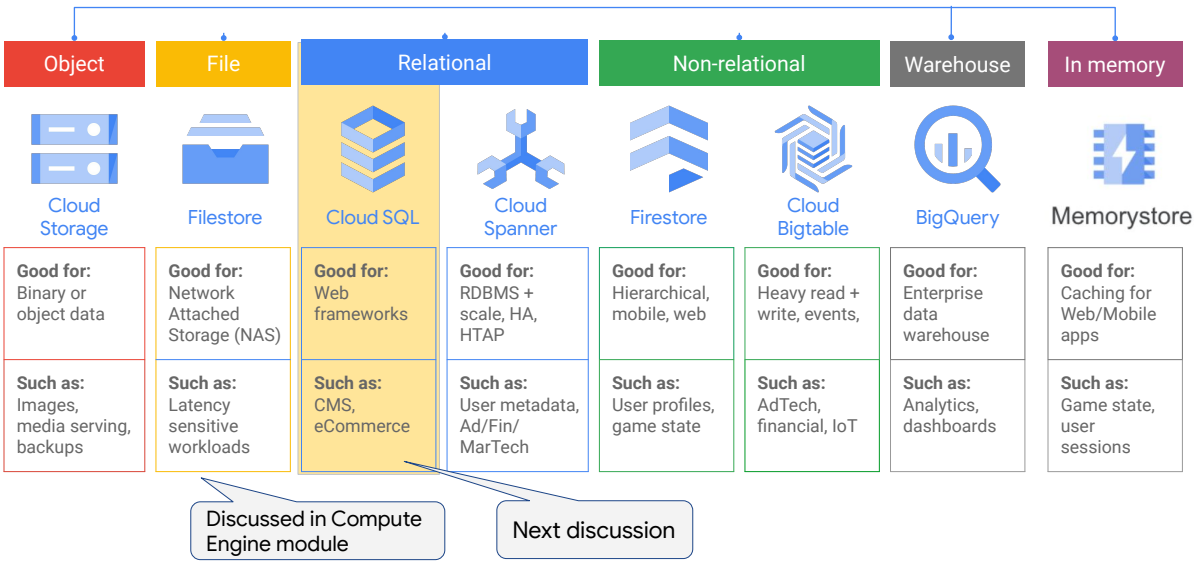
You can manage databases yourself by creating a VM and installing the database software. But you are responsible for everything - backups, patches, OS updates, etc. As an alternative, Google offers fully managed storage options where Google manages the underlying infrastructure

Database Options



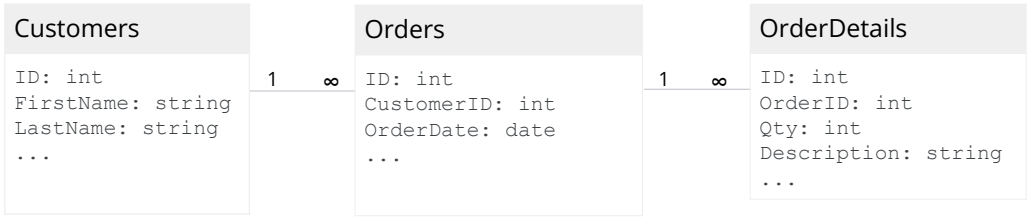
[Your Google Cloud database options, explained]

Storage and database services



What does data in a relational database look like?

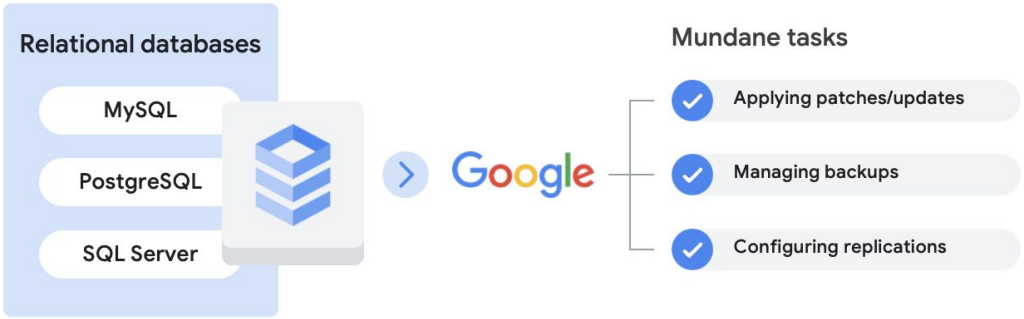
- Tables contain fields, indexes, and constraints
- Primary key ensures each row in a table is unique
- Relationships are constraints that ensure a parent row cannot be deleted if there are child rows in another table



This is a brief (and very, very simplistic) overview of a relational database.

Cloud SQL

A fully managed, cloud-based alternative to on-premise MySQL, PostgreSQL, and SQL Server databases



The business value of Cloud SQL: how companies speed up deployments, lower costs and boost agility

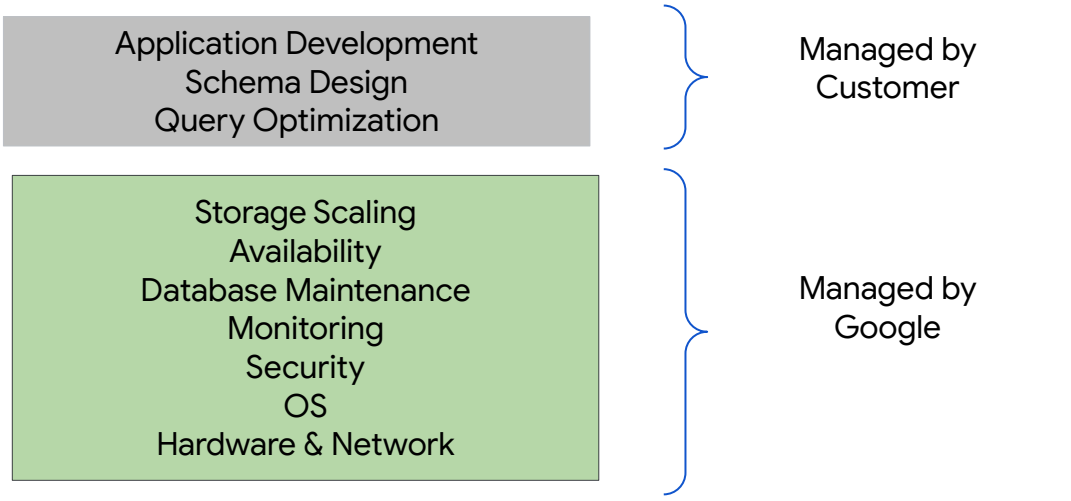
The business value of Cloud SQL: how companies speed up deployments, lower costs and boost agility
https://cloud.google.com/blog/products/databases/the-business-value-of-cloud-sql/

Cloud SQL:
https://cloud.google.com/sql/docs/mysql/introduction

What is Cloud SQL
https://cloud.google.com/blog/topics/developers-practitioners/what-cloud-sql

Cloud SQL offers fully managed relational databases, including MySQL, PostgreSQL, and SQL Server as a service. It's designed to hand off mundane, but necessary and often time-consuming, tasks to Google—like applying patches and updates, managing backups, and configuring replications—so your focus can be on building great applications.

Cloud SQL - shared responsibilities



Google Cloud manages the infrastructure work while the customer focuses on their goals.

Cloud SQL supports a variety of use cases

- Storing and managing relational data such as customer information, product inventory, and financial transactions
- Backend database for web and mobile applications
- Migrating on-premises databases to the cloud
- Building and deploying data-driven applications with minimal setup and management overhead
- Replicating data for disaster recovery and high availability
    - Provides automatic failover for high availability, ensuring that databases are always available in case of an unexpected outage or hardware failure

Cloud SQL - customer use case

- Qlue: Boosting intelligence about activity in Indonesian cities
- Applications include
  - Dashboard to track and resolve incidents of flooding, crime, fire, illegal rubbish dumping, and more
  - Monitor energy usage and traffic congestion in real time
  - Mobile app that enables citizens to report incidents

"If we used a conventional Infrastructure-as-a Service offering, we would have had to dedicate a team to manage operating systems, libraries, services, and load balancing. With Google Cloud, we are able to focus purely on developing our applications and growing the business."

—Surya Darmadi, Chief Operating Officer, Qlue

Qlue: Boosting intelligence about activity in Indonesian cities
https://cloud.google.com/customers/qlue/

Note that Qlue uses multiple Google Cloud services, not just Cloud SQL. It's highly recommended that you understand the use case for all these services, and how they are used together to meet customer needs.

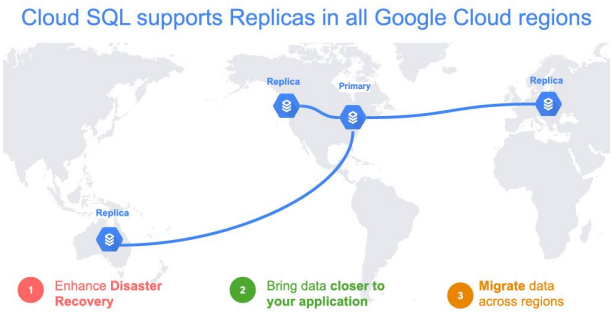
Scaling Cloud SQL Databases

- Cloud SQL can scale in the following ways
  - **Vertical scaling:**
    - Increasing the amount of computing resources (such as CPU and memory) allocated to a single database instance.
    - Can be done with a few clicks in the Cloud Console, and requires no downtime.
  - **Horizontal scaling:**
    - Adding additional read replicas to distribute read workloads and improve performance.
    - Can be added on demand, and there's no limit to the number of replicas that can be added.

Cloud SQL Read Replicas

Fully managed read replica in a different region(s) than that of the primary instance

- Enhance DR
- Bring data closer to your applications (performance and cost implications)
- Migrate data across regions
- Data and other changes on the primary instance are updated in almost real time on the read replicas



Cloud SQL Read Replicas:
https://cloud.google.com/sql/docs/mysql/replication

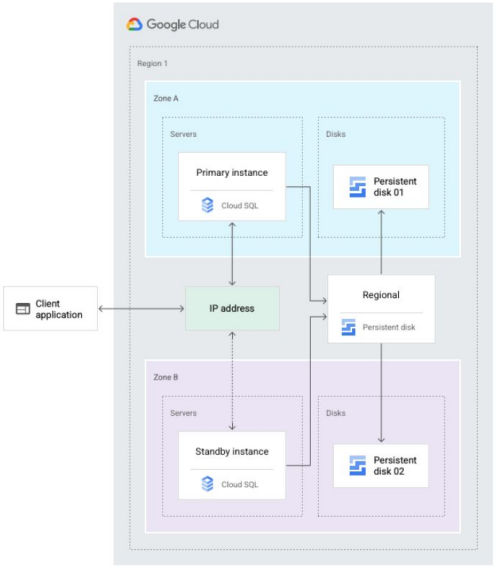
Cloud SQL read replicas provide

- Enhanced Disaster recovery. Can convert a read replica to the Primary (if the primary is no longer available)
- In terms of data closer to your app, you can have different connection strings for an application for different endpoints
    - For example,:
        - Read-Write connection strings will go to the primary
        - Read only connections hit a local read replica
- If reads comprise most of the use case for your app, then accessing the local replica saves on data transfer and egress charges since you are not accessing the primary sitting in the eastern United States in this example

Introducing cross-region replica for Cloud SQL (June 2, 2020)
https://cloud.google.com/blog/products/databases/introducing-cross-region-replica-for-cloud-sql

Cloud SQL - High Availability

- Provides automatic failover if a zone or instance become unavailable
- The primary instance is in one zone in a region
  - The failover instance is in another zone
- Synchronous replication is used to copy all writes from the primary disk to the replica



About high availability
https://cloud.google.com/sql/docs/mysql/high-availability

Connecting and running a query in Cloud SQL

- Connect to Cloud SQL

```
gcloud sql connect myinstancename  --user=root
```

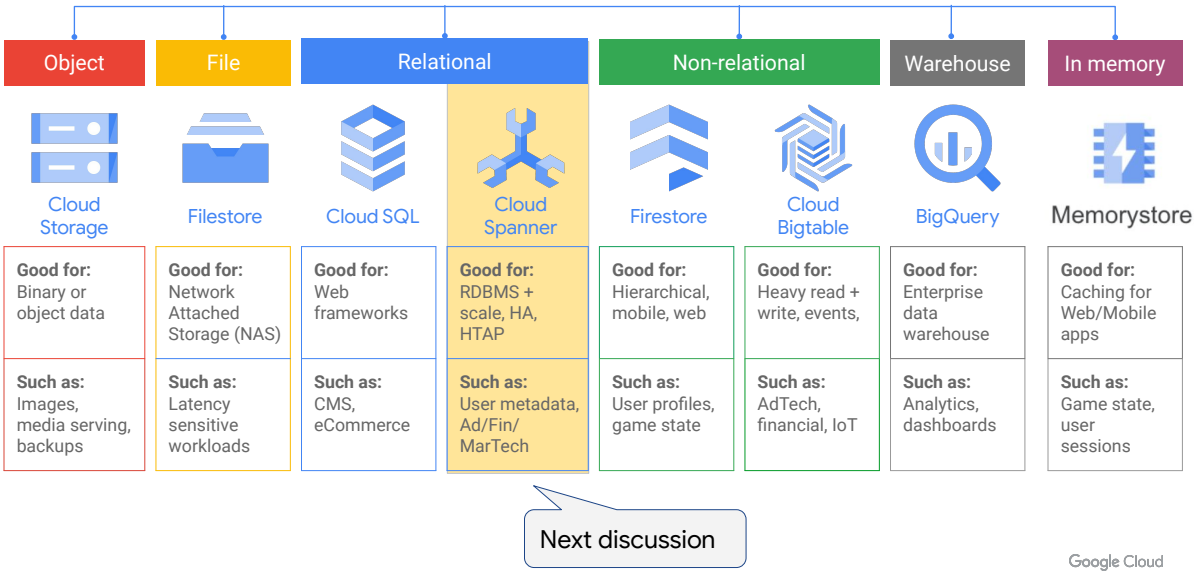
- Execute a query using standard SQL

```
SELECT firstname, lastname, empid FROM employee;
```


Connect to Cloud SQL for MySQL from Cloud Shell
https://cloud.google.com/sql/docs/mysql/connect-instance-cloud-shell

All Cloud SQL for MySQL code samples:
https://cloud.google.com/sql/docs/mysql/samples

Storage and database services



Cloud Spanner

A enterprise-grade, globally distributed, externally consistent relational database having unlimited scalability and industry-leading 99.999% availability

- Powers Google's most popular, globally available products, like YouTube, Drive, and Gmail
- Capable of processing more than 1 billion queries per second at peak
- For any workload, large or small, that cannot tolerate downtime, and requires high availability
- Regional and multi-regional deployments
  - SLA: Multi-regional: 99.999%
  - SLA: Regional: 99.99%
- Supports ANSI standard SQL

Cloud Spanner myths busted

Cloud Spanner:
https://cloud.google.com/blog/topics/developers-practitioners/what-cloud-spanner
https://cloud.google.com/spanner

Cloud Spanner myths busted:
https://cloud.google.com/blog/products/databases/cloud-spanner-myths-busted

Cloud Spanner is an enterprise-grade, globally distributed, externally consistent database that offers unlimited scalability and industry-leading 99.999% availability. It requires no maintenance windows and combines the benefits of relational databases with the unmatched scalability and availability of non-relational databases.

Spanner powers Google's most popular, globally available products, like YouTube, Drive, and Gmail, and it can processes more than 1 Billion queries per second at peak

But don't be mislead into thinking that Spanner is only for large, enterprise level applications. Many customers use Spanner for their smaller workloads (both in terms of transactions per second and storage size) for availability and scalability reasons. Spanner is appropriate for any customer that cannot tolerate downtime, and needs high availability for their applications

Limitless scalability and high availability is critical in many industry verticals such as gaming and retail, especially when a newly launched game goes viral and becomes

an overnight success or when a retailer has to handle a sudden surge in traffic due to a Black Friday/Cyber Monday sale.

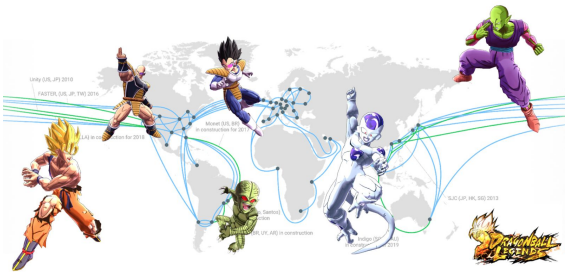
Spanner offers the flexibility to interact with the database via a SQL dialect based on ANSI 2011 standard as well as via a REST or gRPC API interface, which are optimized for performance and ease-of-use. In addition to Spanner's interface, there is a PostgreSQL interface for Spanner, that leverages the ubiquity of PostgreSQL and provides development teams with an interface that they are familiar with. The PostgreSQL interface provides a rich subset of the open-source PostgreSQL SQL dialect, including common query syntax, functions, and operators. It also supports a core collection of open-source PostgreSQL data types, DDL syntax, and information schema views. You get the PostgreSQL familiarity, and relational semantics at Spanner scale.

Cloud Spanner supports a variety of use cases

- Large-scale, multi-regional data storage
  - Can store and manage terabytes or petabytes of data across multiple regions
- Online transaction processing (OLTP) applications:
  - Can support OLTP workloads with low latency and high throughput, making it suitable for applications that require real-time data processing, such as e-commerce or financial services
- Banking and financial services
  - Spanner's high-availability and consistency guarantees make it well-suited for use cases in the financial services industry, such as stock trading or payment processing
- Geographically distributed data management
  - Spanner supports multiple geographic locations and provides a globally consistent view of data, making it ideal for use cases that require a database that can handle data distributed across multiple regions or continents

Cloud Spanner - customer use case

- Dragon Ball Legends - mobile game from Bandai Namco Entertainment
- Requirements were:
  - Global backend that could scale with millions of players and still perform well.
  - Global reliable, low latency network to support multi-region player-versus-player battles
  - Real-time data analytics to measure and evaluate how people are playing the game and adjust it on-the-fly.



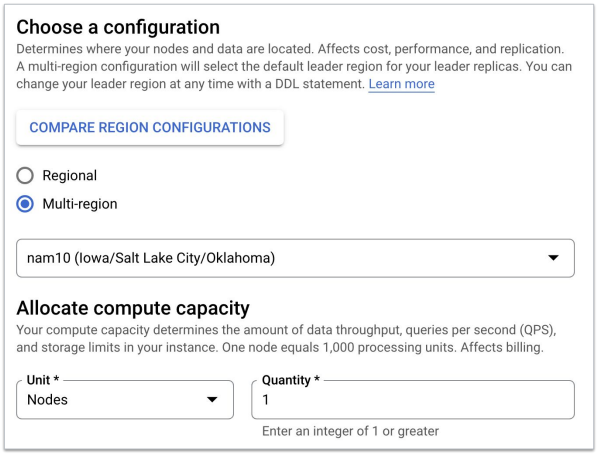
Behind the scenes with the Dragon Ball Legends GC backend

This is another example where multiple Google Cloud services are used together to provide a solution.

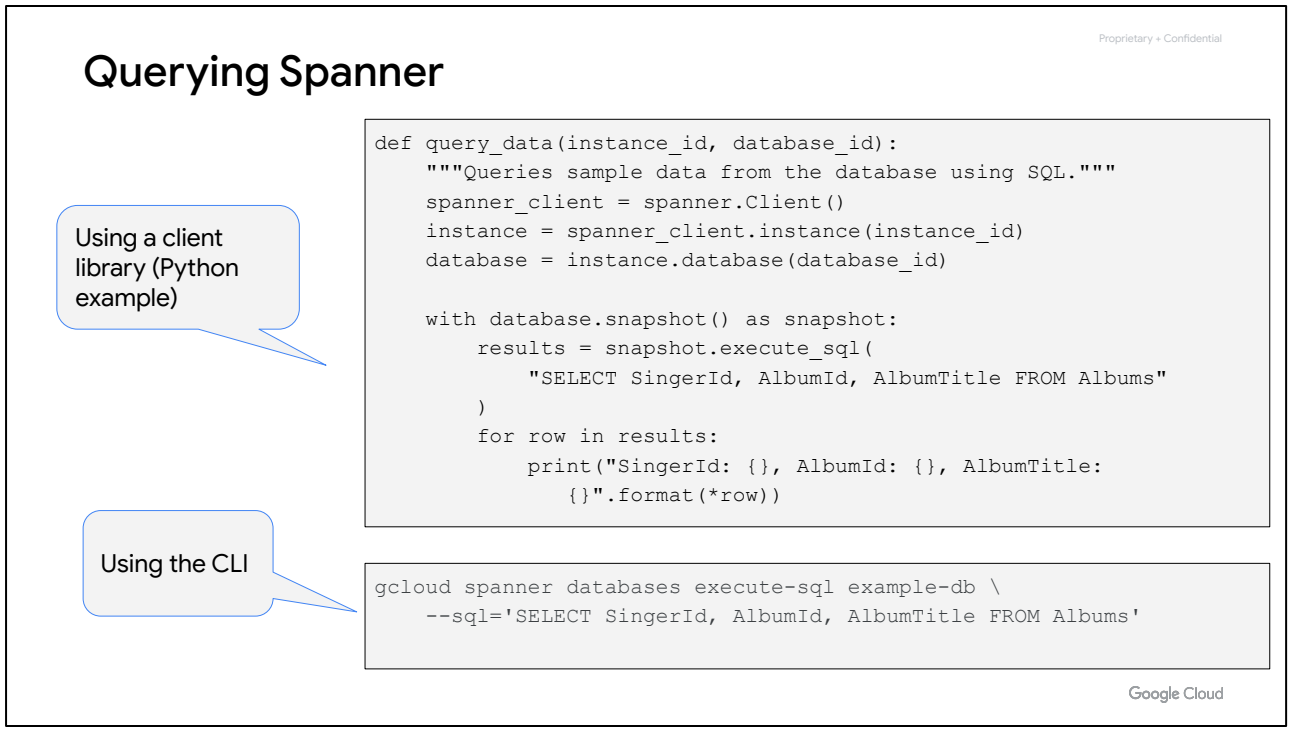
Scaling Cloud Spanner

Scales out (horizontal scaling)

- Manually add nodes/processing units to support more data and users as needed
- Turn on autoscaling to automatically adjust the number of nodes in an instance to handle changing traffic patterns and load



Autoscaling Cloud Spanner
https://cloud.google.com/architecture/autoscaling-cloud-spanner



Query syntax in Google Standard SQL
https://cloud.google.com/spanner/docs/reference/standard-sql/query-syntax

Python quickstart
https://cloud.google.com/spanner/docs/getting-started/python

Spanner code examples:
https://cloud.google.com/spanner/docs/samples

If you already know database programming, you will be comfortable using Spanner. Like all relational databases, you create tables, tables have fields, fields have data types and constraints. You can set up relationships between tables. When you want to store data, you add rows to tables.

Once you have data, you can retrieve it with a standard SQL SELECT statement.

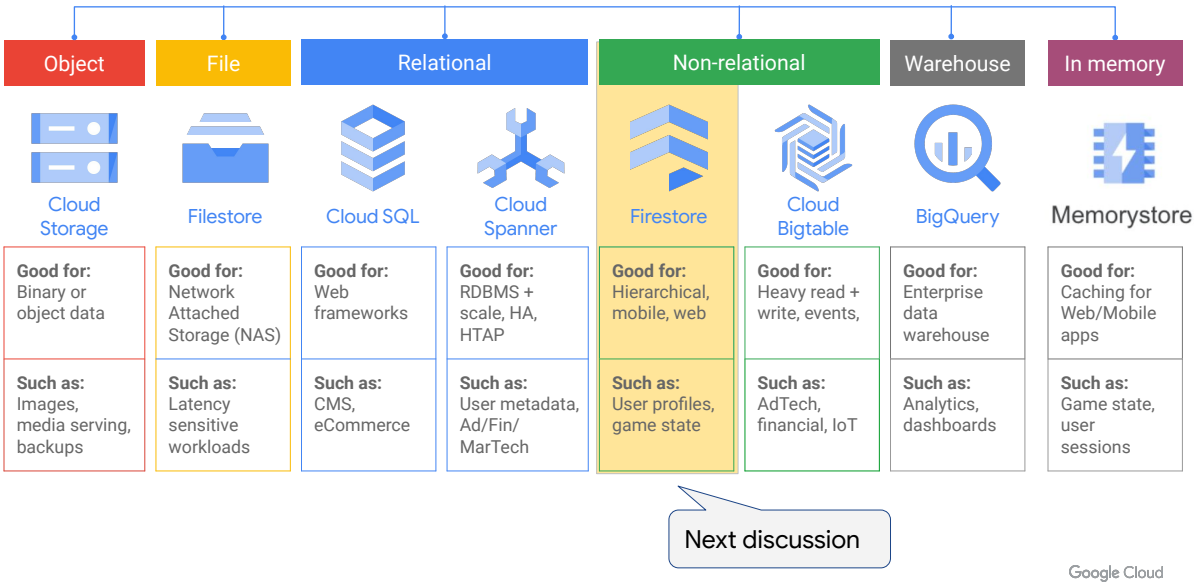
Cloud SQL

- Max 64TB data
- High Availability via master / standby
- 99.95% SLA
- Vertically Scalable via reprovisioning
- Horizontally Scalable by read replicas
- Planned maintenance windows

Cloud Spanner

- 4TB / node data
- Distributed service - always available
- 99.99% SLA (regional)
- 99.999% SLA (multi-region)
- Horizontally Scalable
- No maintenance managed service

Storage and database services



Types of NoSQL Database

- Key-value stores (**Cloud Memorystore**)
  - Data is stored in key-value pairs
  - Examples include Redis and SimpleDB
- Document stores (**Cloud Firestore**)
  - Data is stored in some standard format like XML or JSON
  - Nested and hierarchical data can be stored together
  - MongoDB, CouchDB, and DynamoDB are examples
- Wide-column stores (**Cloud Bigtable**)
  - Key identifies a row in a table
  - Columns can be different within each row
  - Cassandra and HBase are examples

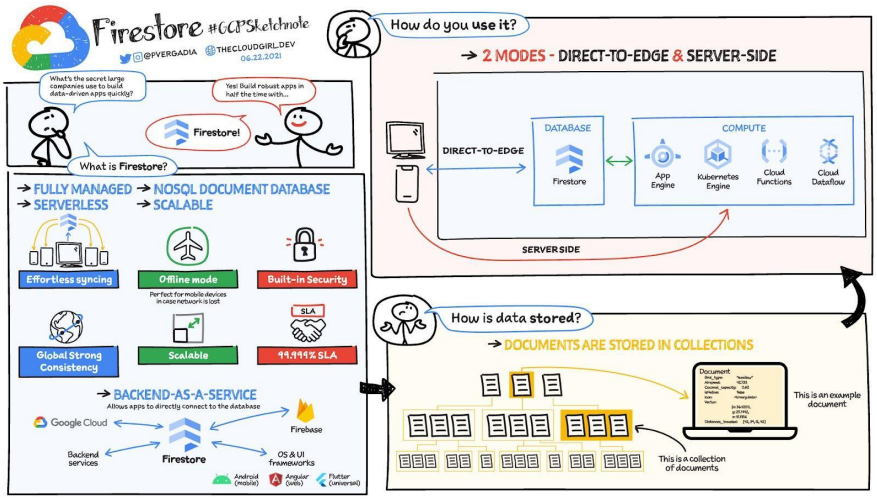
There are different types of NoSQL databases. (The Google Cloud services are in bold text in the above slide.)

Key-value stores are the simplest. These act like a map or dictionary. To save data, just specify a key and assign a value to that key.

Document stores allow you to store hierarchical data. So, instead of having an orders table and a details table, you can store an order in a single document. That document can have properties that represent the order along with an array of subdocuments representing the details. The data can be stored in the database as JSON or XML, or a binary format called BSON.

Wide-column stores have data in tables. Each row in the table has a unique value or key. Associated with that key can be any number of columns. Each column can have any type of data. There's no schema, so different rows in the same table can have different columns.

Cloud Firestore



[All you need to know about Firestore]

All you need to know about Firestore:
https://cloud.google.com/blog/topics/developers-practitioners/all-you-need-know-about-firestore-cheatsheet

Cloud Firestore

- Completely managed, document store, NoSQL Database
  - No administration, no maintenance, nothing to provision
- 1 GB per month free tier
- Indexes created for every property by default
  - Secondary indexes and composite indexes are supported
- Supports ACID* transactions
- For mobile, web, and IoT apps at global scale
  - Live synchronization and offline support
- Multi-region replication

*ACID explained: https://en.wikipedia.org/wiki/ACID

Firestore:
https://cloud.google.com/firestore/docs/

Cloud Firestore is a fast, fully managed, serverless, cloud-native NoSQL document database

It simplifies storing, syncing, and querying data for your mobile, web, and IoT apps at global scale

Its client libraries provide live synchronization and offline support, and its security features and integrations with Firebase and Google Cloud accelerate building truly serverless apps.

Cloud Firestore also supports ACID transactions, so if any of the operations in the transaction fail and cannot be retried, the whole transaction will fail.

Also, with automatic multi-region replication and strong consistency, your data is safe and available, even when disasters strike.

Cloud Firestore allows you to run sophisticated queries against your NoSQL data without any degradation in performance. This gives you more flexibility in the way you structure your data.

Some features recently introduced are

- VPC Service Controls for Firestore. This allows you to define a perimeter to mitigate data exfiltration risks.
- Firestore triggers for Cloud Functions. When certain events happen in Firestore, Cloud Functions can run in response.
- The same Key Visualizer service that's available for Bigtable and Spanner is also available for Firestore. It allows developers to quickly and visually identify performance issues

Firestore example data

An index is created for every property so that queries are extremely fast

Collections

Documents live in collections, which are simply containers for documents. For example, you could have a `users` collection to contain your various users, each represented by a document:

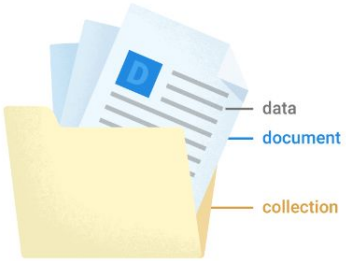
users

alovelace

```
first : "Ada"
last : "Lovelace"
born : 1815
```

aturing

```
first : "Alan"
last : "Turing"
born : 1912
```

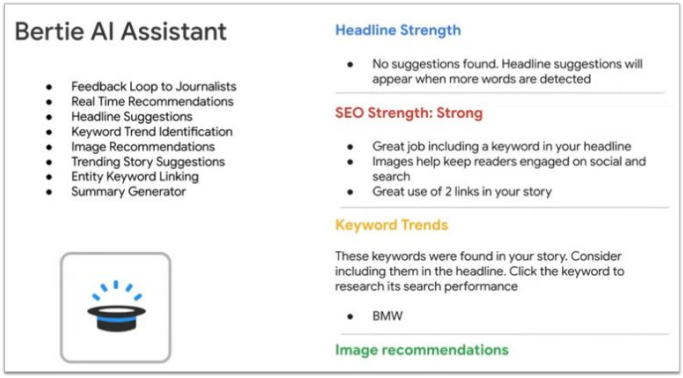


Cloud Firestore is schemaless, so you have complete freedom over what fields you put in each document and what data types you store in those fields. Documents within the same collection can all contain different fields or store different types of data in those fields. However, it's a good idea to use the same fields and data types across multiple documents, so that you can query the documents more easily.

From: https://firebase.google.com/docs/firestore/data-model#hierarchical-data

Firestore - customer use case

- Forbes created Bertie - an AI assistant for journalists
- Journalists upload their content and Bertie provides feedback
  - Strength of the article's headline
  - Keywords needed for search engine optimization
  - Words to add to the headline to increase search performance

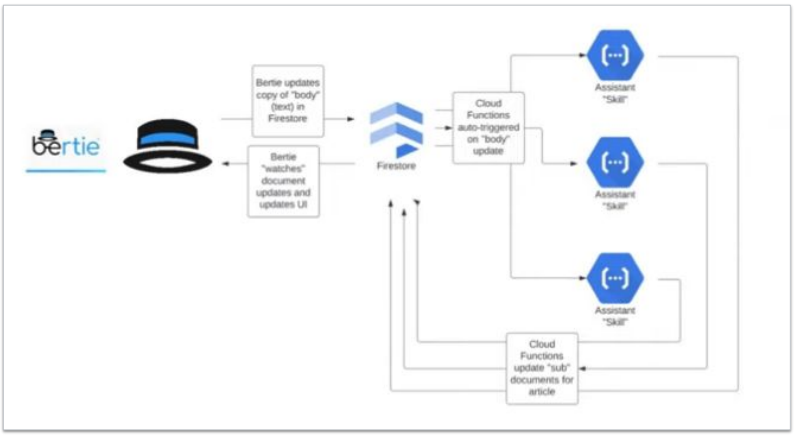


YouTube video: https://www.youtube.com/watch?v=KVRxsRPhmoo

Forbes website (for anyone that wants to know more about the company) https://www.forbes.com/

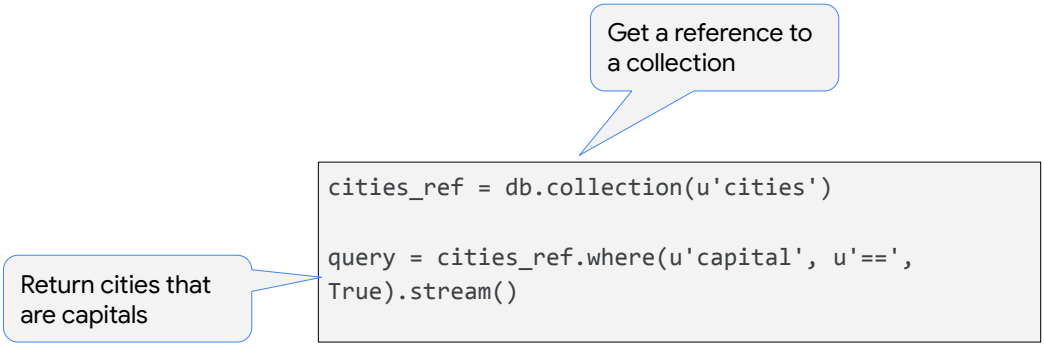
Forbes - Bertie Architecture

- Content is stored in Firestore
- Data updates trigger Cloud Functions
  - Each Cloud Function performs a different task
  - Results are written back to Firestore
  - Website is refreshed with the recommendations



YouTube video: https://www.youtube.com/watch?v=KVRxsRPhmoo

Querying Firestore - Python example



Choosing between Native mode and Datastore mode
https://cloud.google.com/datastore/docs/firestore-or-datastore

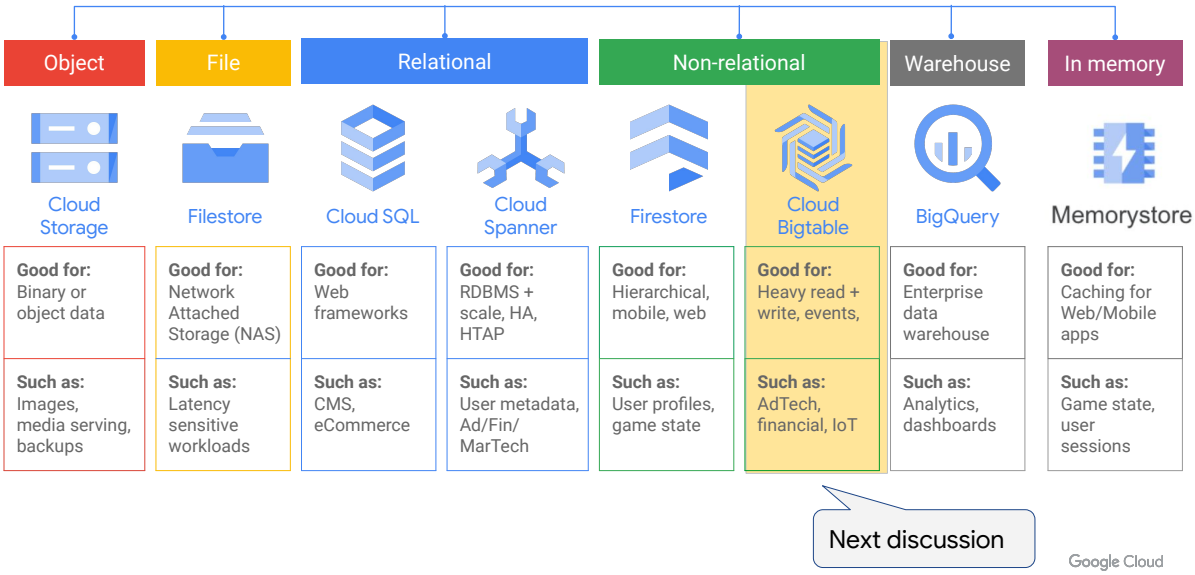
Firestore in Datastore Mode Queries
https://cloud.google.com/datastore/docs/concepts/queries

Firestore in Native Mode Queries
https://cloud.google.com/firestore/docs/samples

Firestore code examples:
https://cloud.google.com/firestore/docs/samples

Firestore - Querying and filtering data:
https://cloud.google.com/firestore/docs/query-data/queries

Storage and database services



Cloud Bigtable

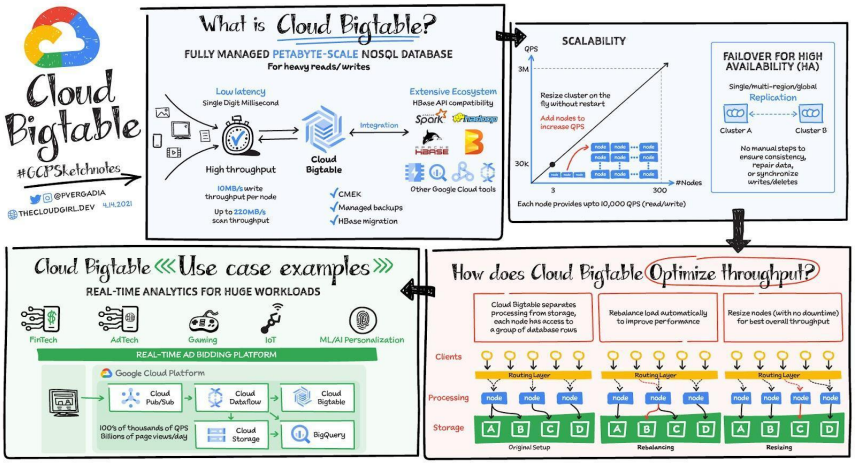
Highly scalable NoSQL database that can handle billions of rows and petabytes of data, making it ideal for use cases that require large-scale data storage and processing with up to 99.999% availability

- High throughput at low latency
  - Ideal for use cases that require real-time data processing and analysis
- Integrates easily with big data tools
  - Same API as HBase
  - Allows on-premises HBase applications to be easily migrated
- Consistent sub-10ms latency

Bigtable:
https://cloud.google.com/bigtable

- Bigtable is a fully managed, scalable NoSQL database service for large analytical and operational workloads with up to 99.999% availability
- Bigtable is built with proven infrastructure that powers Google products used by billions such as Search and Maps.
- Bigtable is ideal for storing very large amounts of data in a key-value store and supports high read and write throughput at low latency for fast access to large amounts of data.
- It is a Fully managed service that integrates easily with big data tools like Hadoop, Dataflow, and Dataproc. Plus, support for the open source HBase API standard makes it easy for development teams to get started
- Some of the features are
  - Autoscaling: Autoscaling helps prevent over-provisioning or under-provisioning by letting Cloud Bigtable automatically add or remove nodes to a cluster when usage changes.
    - In addition, metrics are available to help you understand how autoscaling is working.
  - You can use customer managed encryption keys (CMEK) in Cloud Bigtable instances, including ones that are replicated across multiple regions.
  - App profile cluster groups let you route an app profile's traffic to a subset of an instance's clusters.

Cloud Bigtable



How BIG is Cloud Bigtable?

How BIG is Cloud Bigtable:
https://cloud.google.com/blog/topics/developers-practitioners/how-big-cloud-bigtable

Cloud Bigtable example use cases

Healthcare	Patient data and other health-related information
Financial Services	Storage of large amounts of transaction, risk management, and compliance data
Retail	Storage of data related to customer behavior which can be used to generate product recommendations
Gaming	Player data for game analytics
IoT	Data generated by connected devices and sensors
Logs and Metrics	Storage of log data and metrics for analysis
Time series data	Storage of resource consumption like CPU and memory usage over time for multiple servers

Cloud Bigtable example data

One key column

Max size of column is 256 MB/ea

Data that's likely to be accessed via the same request are grouped into column families

Column Families

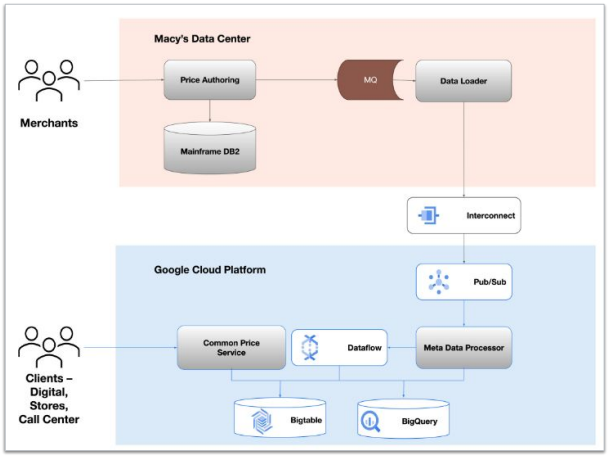
Row Key	Flight_Information					Aircraft_Information			
	Origin	Destination	Departure	Arrival	Passengers	Capacity	Make	Model	Age
ATL#arrival#20190321-1121	ATL	LON	20190321-0311	20190321-1121	158	162	B	737	18
ATL#arrival#20190321-1201	ATL	MEX	20190321-0821	20190321-1201	187	189	B	737	8
ATL#arrival#20190321-1716	ATL	YVR	20190321-1014	20190321-1716	201	259	B	757	23

Rows can have millions of columns

Cloud Bigtable provides Column Families. By accessing the Column Family, you can pull some of the data you need without pulling all of the data from the row or having to search for it and assemble it. This makes access more efficient.

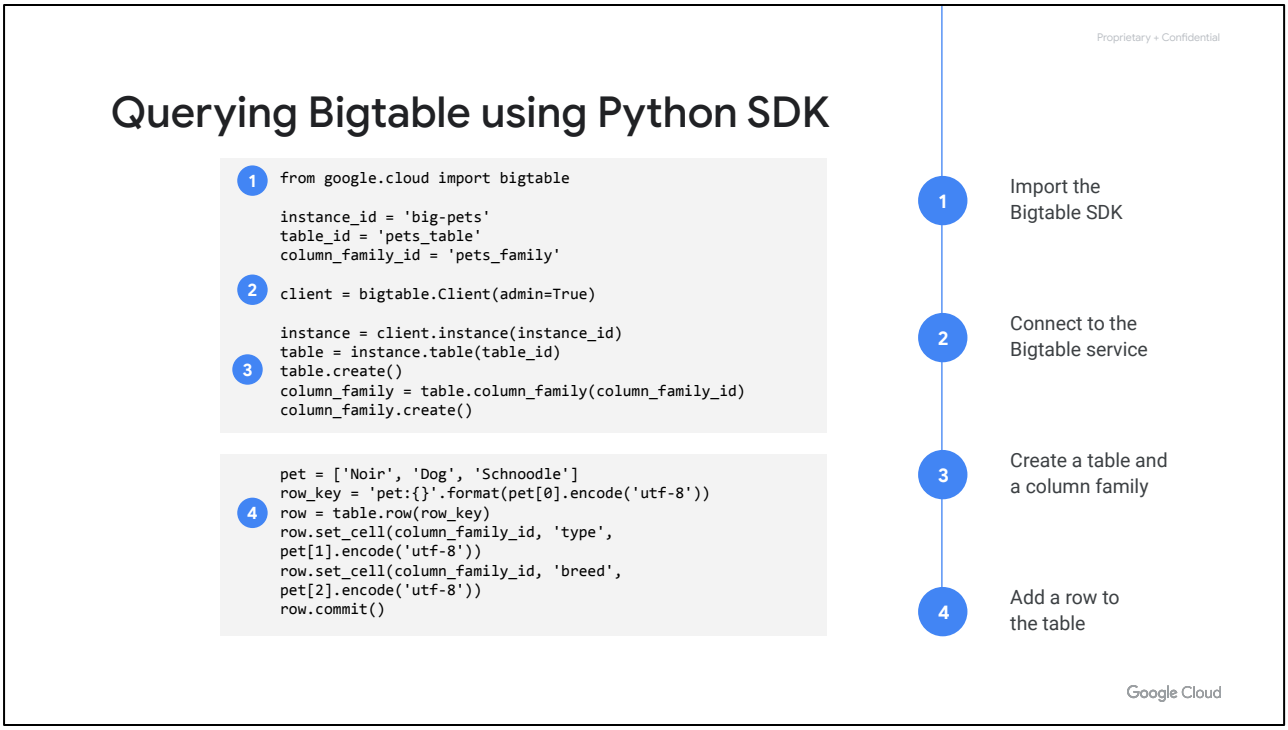
Bigtable - customer use case

- Macy's sells a wide range of merchandise, including apparel and accessories (men's, women's and children's), cosmetics, home furnishings and other consumer goods
  - 700+ stores across the US
- Bigtable provides data for the pricing system
  - Access pattern entails finding an item's ticket price based on a given division, location, and the universal price code
  - Can search millions of product codes within single digit milliseconds



How Macy's enhances the customer experience with Google Cloud services

How Macy's enhances the customer experience with Google Cloud services
https://cloud.google.com/blog/products/databases/how-macys-enhances-customer-experience-google-cloud-services



Python Client for Google Cloud Bigtable
https://googleapis.dev/python/bigtable/latest/index.html

Here's some Python code that uses the Bigtable SDK. The takeaway should be that the code is pretty simple. Connect to the database, create a table, tables have column families. You can then add rows. Rows require a unique ID or row key. Rows have columns that are in column families.

Querying Bigtable (continued)

1
```
key = 'pet:Noir'.encode('utf-8')
row = table.read_row(key)
breed =
row.cells[column_family_id]['breed'][0].value.decode('utf-8')
type =
row.cells[column_family_id]['type'][0].value.decode('utf-8')
```

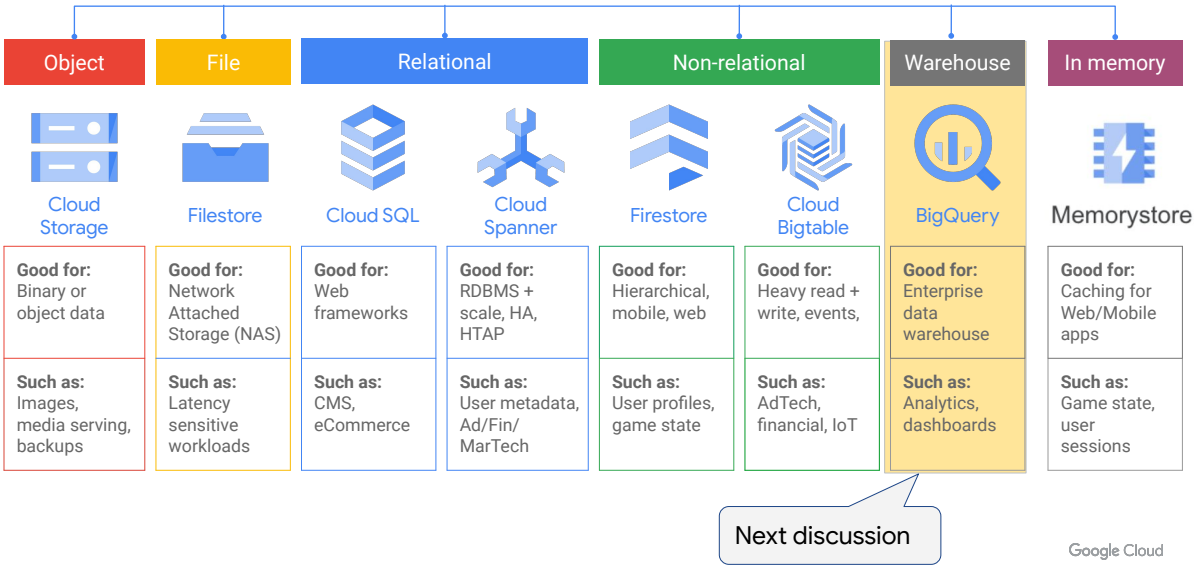
2
```
results = table.read_rows()
results.consume_all()

for row_key, row in results.rows.items():
    key = row_key.decode('utf-8')
    type =
row.cells[column_family_id]['type'][0].value.decode('utf-8')
    breed =
row.cells[column_family_id]['breed'][0].value.decode('utf-8')
```

1 Retrieve a row

2 Retrieve many rows

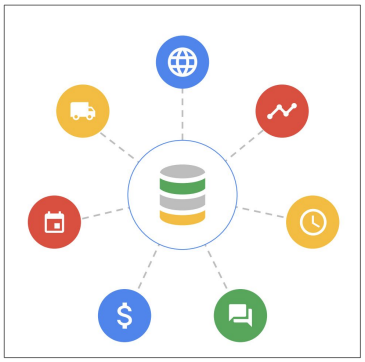
Once you have some data, you can read individual or multiple rows. Remember, to get high performance, you want to retrieve rows using the row key.

Storage and database services



What is a Data Warehouse?

- Allows data from multiple sources to be combined and analyzed
  - Historical archive of data
- Data sources could be:
  - Relational databases
  - Logs
  - Web data
  - Etc.
- Optimized for analytical processing
  - Can handle large amounts of data and complex queries, and is well-suited for reporting and data analysis



A data warehouse is a central hub for business data. Different types of data can be transformed and consolidated into the warehouse for analysis

BigQuery

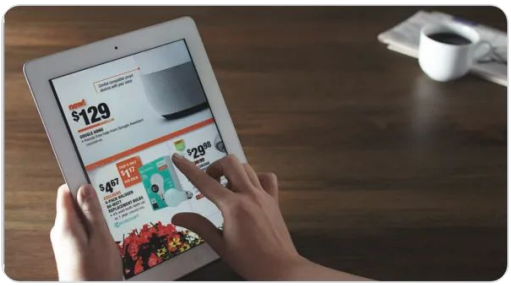
- Fully managed, serverless, highly scalable data warehouse
- Multi-cloud capabilities using standard SQL
- Processes multi-terabytes of data in minutes
- Automatic high availability
- Supports federated queries
  - Cloud SQL & Cloud Spanner
  - Cloud Bigtable
  - Files in Cloud Storage
- Use cases:
  - Near real-time analytics of streaming data to predict business outcomes with built-in machine learning, geospatial analysis and more
  - Analysis of historical data

BigQuery

Cloud data warehouse to power your data-driven innovation
https://cloud.google.com/bigquery

BigQuery - customer use case

- The Home Depot is the world's largest home improvement retailer
  - 2,300 stores in North America + online retail
  - Annual sales > $100 billion
- BigQuery provides timely data to help keep 50,000+ items stocked at over 2,000 locations, to ensure website availability, and provide relevant information through the call center
- No two Home Depots are alike, and the stock in each has to be managed at maximum efficiency.
  - Migrating to Google Cloud, THD's engineers built one of the industry's most efficient stock replenishment systems

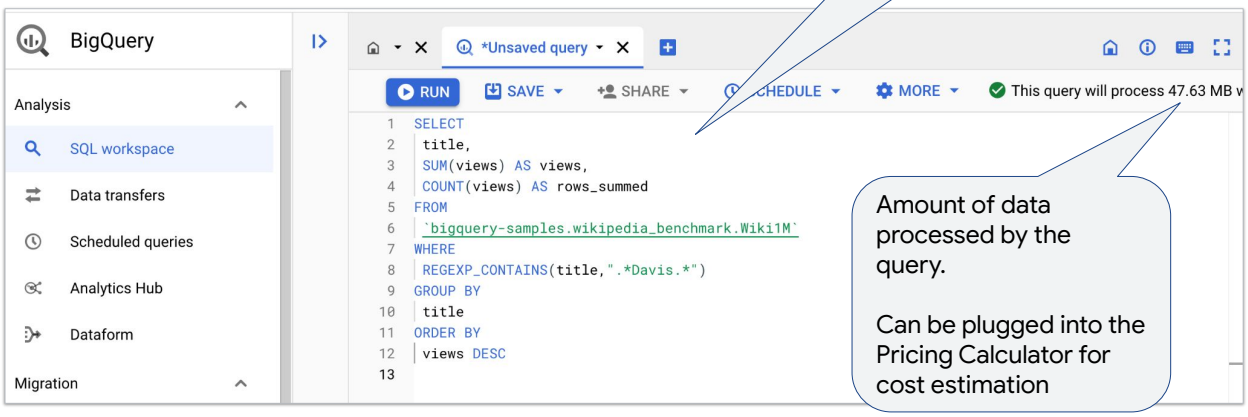


The Home Depot's data-driven focus on customer success

About The Home Depot
https://corporate.homedepot.com/page/about-us

The Home Depot's data-driven focus on customer success
https://cloud.google.com/customers/featured/the-home-depot

BigQuery: Executing queries in the console



Run interactive and batch query jobs
https://cloud.google.com/bigquery/docs/running-queries#bigquery-query-cl

BigQuery: Executing queries in the CLI

```
bq query --use_legacy_sql=false \
'SELECT
  word,
  SUM(word_count) AS count
FROM
`bigquery-public-data`.samples.shakespeare
WHERE
  word LIKE "%raisin%"
GROUP BY
  word'
```


bq command line tool:
https://cloud.google.com/bigquery/docs/bq-command-line-tool

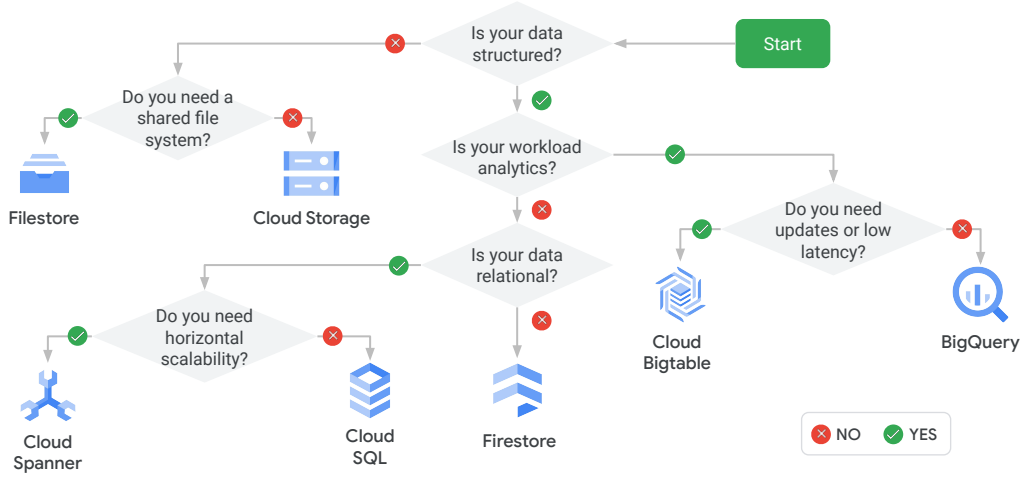
BigQuery: Executing queries using client library

Python example

```
from google.cloud import bigquery

# Construct a BigQuery client object.
client = bigquery.Client()

query = """
    SELECT name, SUM(number) as total_people
    FROM `bigquery-public-data.usa_names.usa_1910_2013`
    WHERE state = 'TX'
    GROUP BY name, state
    ORDER BY total_people DESC
    LIMIT 20
"""
query_job = client.query(query)  # Make an API request.

print("The query data:")
for row in query_job:
    # Row values can be accessed by field name or index.
    print("name={}, count={}".format(row[0], row["total_people"]))
```


All BigQuery code samples
https://cloud.google.com/bigquery/docs/samples

Storage and database decision chart



Design an optimal storage strategy for your cloud workload

Design an optimal storage strategy for your cloud workload
https://cloud.google.com/architecture/storage-advisor

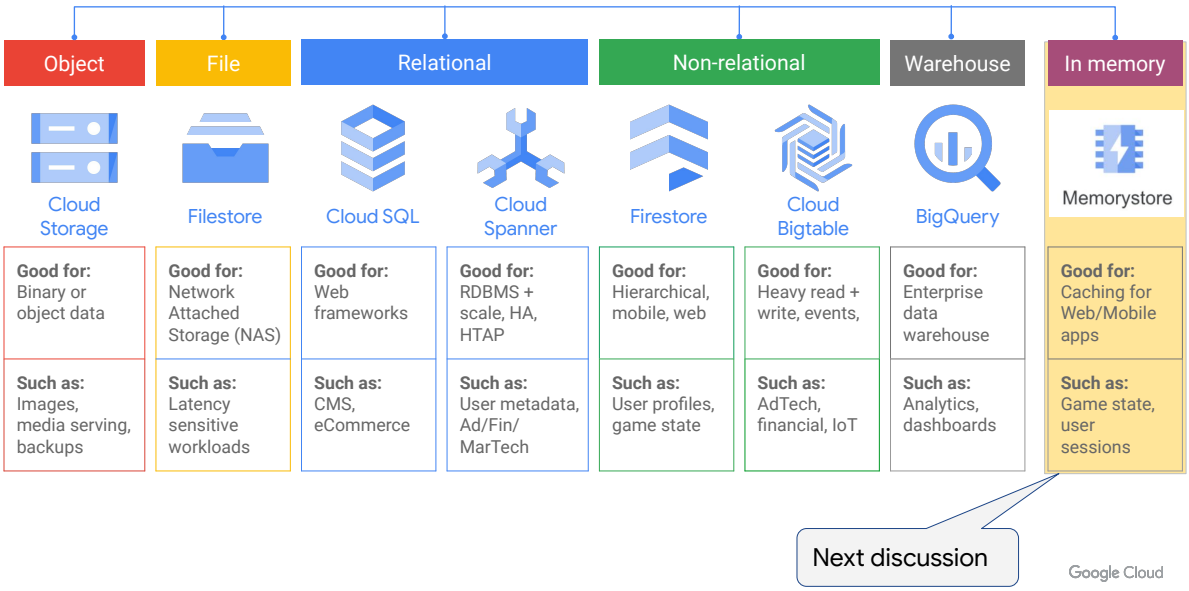
Let's summarize the services in this module with this decision chart:

- First, ask yourself: Is your data structured, and will it need to be accessed using its structured data format? If the answer is no, then ask yourself if you need a shared file system. If you do, then choose Filestore.
- If you don't, then choose Cloud Storage.
- If your data is structured and needs to be accessed in this way, than ask yourself, does your workload focus on analytics? If it does, you will want to choose Cloud Bigtable or BigQuery, depending on your latency and update needs.
- Otherwise, check whether your data is relational. If it's not relational, choose Firestore.
- If it is relational, you will want to choose Cloud SQL or Cloud Spanner, depending on your need for horizontal scalability.

Depending on your application, you might use one or several of these services to get the job done. For more information on how to choose between these different services, please refer to the following two links:

https://cloud.google.com/storage-options/
https://cloud.google.com/products/databases/

Storage and database services



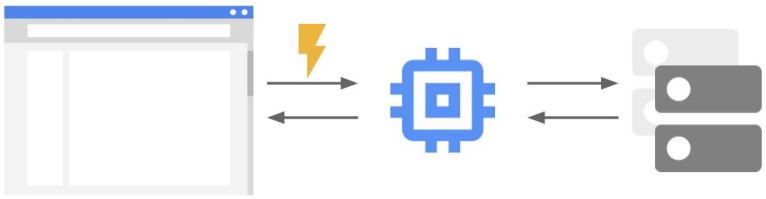
Memorystore

- Fully managed implementation of the open source in-memory databases Redis and Memcached
- High availability, failover, patching and monitoring
- Sub-millisecond latency
- Instances up to 300 GB
- Network throughput of 12 Gbps
- Use cases:
  - Lift and shift of Redis, Memcached
  - Anytime need a managed service for cached data

Memorystore

Memorystore
https://cloud.google.com/memorystore

In-memory caching



Application

Web browser

Cache

A caching layer is a high-speed storage layer that stores a subset of data.

Databases

Benefit

- Reduce latency
- Reduce back-end load

Main use case

- Gaming leaderboard
- Real-time application
- Social Media

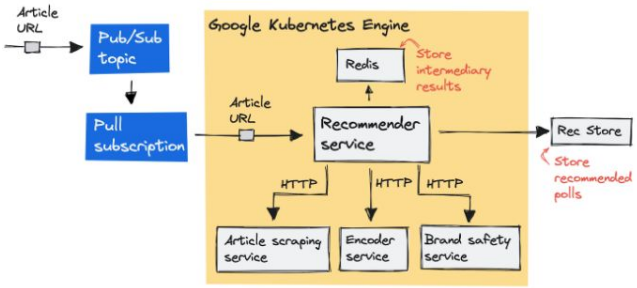
```
gcloud redis instances create my-redis-instance
gcloud memcache instances create my-memcache-instance
```

When to use Cloud Memorystore

Application domain	Game	Social Media	IoT	Analytics
Cloud Memorystore -driven capabilities	Leaderboard, advertisement	Chat, news feed	Real-time transactions, sensor analytics	Read-heavy dashboard

Memorystore - customer use case

- Opinary creates polls that appear alongside news articles on various sites around the world
  - Machine learning is used to decide which poll to display by which article
- The polls let users share their opinion with one click and see how they compare to other readers.
- Publishers benefit by increased reader retention, and increased subscriptions
- Advertisers benefit from high-performing interaction with their audiences

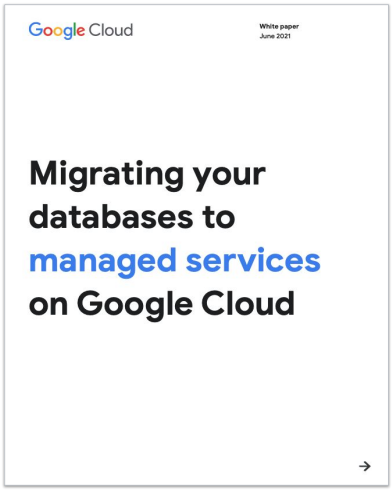


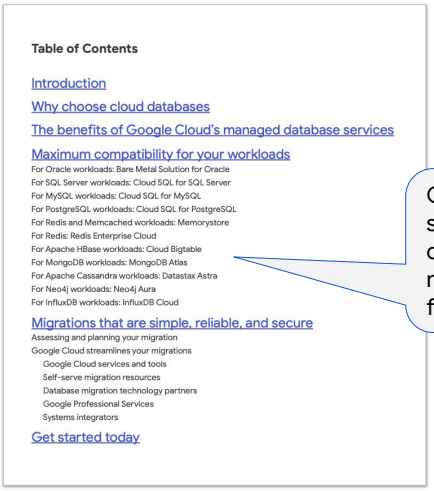
Opinary generates recommendations faster on Cloud Run

Opinary generates recommendations faster on Cloud Run
https://cloud.google.com/blog/topics/developers-practitioners/opinary-generates-recommendations-faster-cloud-run/

More about Opinary (if interested)
https://opinary.com/

Google white paper - database migration

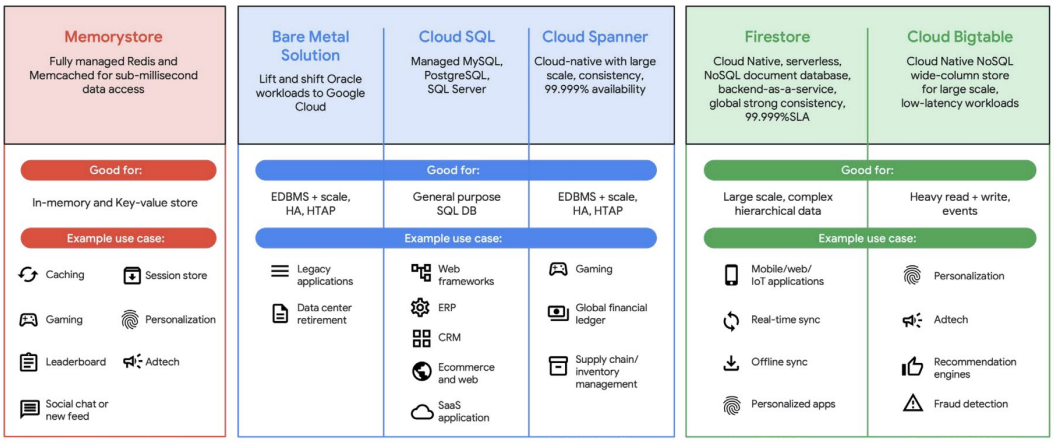




Covers the services discussed in this module plus a few more

Migrating Databases

Which database should I use?



Make your database your secret advantage

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

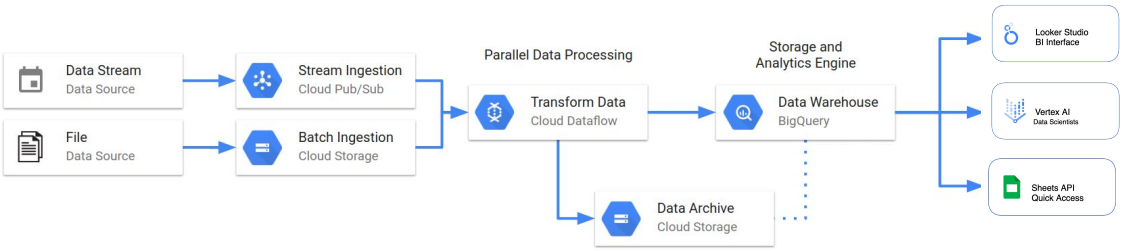
3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore , Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

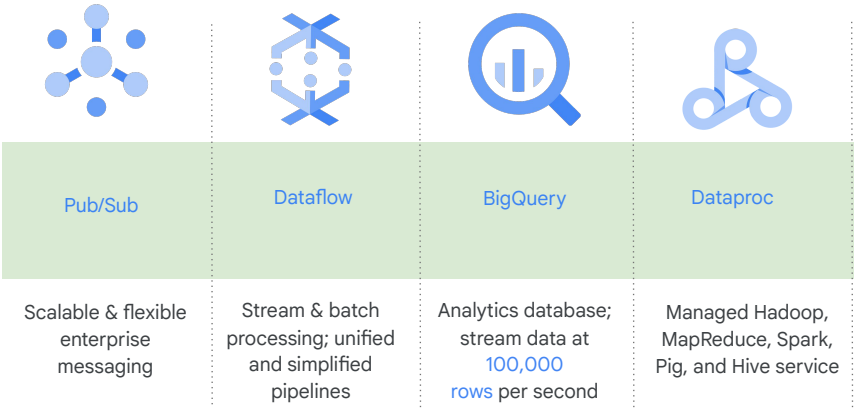
Typical Data Processing Pipeline



How to Build a data pipeline with Google Cloud

How to Build a data pipeline with Google Cloud
https://www.youtube.com/watch?v=yVUXvabnMRU

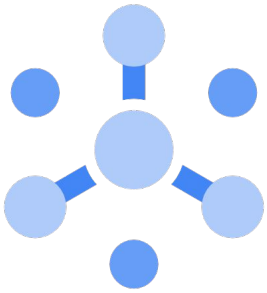
Google Cloud big data services are fully managed and scalable



Google Cloud Big Data solutions are designed to help you transform your business and user experiences with meaningful data insights. It is an integrated, serverless platform. “Serverless” means you don’t have to provision compute instances to run your jobs. The services are fully managed, and you pay only for the resources you consume. The platform is “integrated” so Google Cloud data services work together to help you create custom solutions.

Pub/Sub is scalable, reliable messaging

- Fully managed, massively scalable messaging service
  - It allows messages to be sent between independent applications
  - Can scale to millions of messages per second
- Messages are sent and received via HTTP(S)
- Supports multiple senders and receivers simultaneously
- Global service
  - Messages are copied to multiple zones for greater fault tolerance
  - Dedicated resources in every region for fast delivery worldwide
- Pub/Sub messages are encrypted at rest and in transit



What is Cloud Pub/Sub?

What is Cloud Pub/Sub?
https://www.youtube.com/watch?v=JrKEErlWvzA&list=PLTWE_Imu2InBzuPmOcgAYP7U80a87cpJd

Cloud Pub/Sub is a fully managed, massively scalable messaging service that can be configured to send messages between independent applications, and can scale to millions of messages per second.

Pub/Sub messages can be sent and received via HTTP and HTTPS.

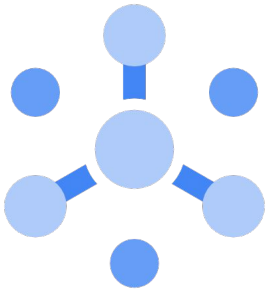
It also supports multiple senders and receivers simultaneously.

Pub/Sub is a global service. Fault tolerance is achieved by copying the message to multiple zones and using dedicated resources in every region for fast worldwide delivery.

All Pub/Sub messages are encrypted at rest and in transit.

Why use Pub/Sub?

- Building block for data ingestion in Dataflow, Internet of Things (IoT), Marketing Analytics, etc.
- Provides push notifications for cloud-based applications.
- Connects applications across Google Cloud (push/pull between components (e.g. GCE and App Engine)



Pub/Sub is an important building block for applications where data arrives at high and unpredictable rates, like Internet of Things systems. If you're analyzing streaming data, Dataflow is a natural pairing with Pub/Sub.

Pub/Sub also works well with applications built on Google Cloud's compute platforms. You can configure your subscribers to receive messages on a "push" or a "pull" basis. In other words, subscribers can get notified when new messages arrive for them, or they can check for new messages at intervals.

Pub/Sub - customer use case

- Sky is one of Europe's leading media and communications companies, providing Sky TV, streaming, mobile TV, broadband, talk, and line rental services to millions of customers in seven countries
- **Pub/Sub** is used to stream diagnostic data from millions of Sky Q TV boxes
- Data is then parsed through **Cloud Dataflow** to **Cloud Storage** and **BigQuery**, monitored on its way by Stackdriver (**Operations**), which triggers email and Slack alerts should issues occur.

"This fully functioning Google Cloud solution will act as a blueprint for future Sky projects. We can capture all diagnostic data in Google Cloud and use it to inform our future strategy. Sky management sees this as the beginning of a new era in data management, analytics, and data science."

—Oliver Tweedie, Director of Data Engineering, Sky

Sky: Scaling for success with Sky Q diagnostics

Dataproc is managed Hadoop

- Fast, easy, managed way to run Hadoop and Spark/Hive/Pig on Google Cloud
- Create clusters in 90 seconds or less on average
- Scale clusters up and down even when jobs are running
- Dataproc storage
  - Automatically installs the HDFS-compatible Cloud Storage connector
    - Run Apache Hadoop or Apache Spark jobs directly on data in Cloud Storage
  - Alternatively can use boot disks to store data
    - Deleted when the Dataproc cluster is deleted

Overview
https://cloud.google.com/dataproc

Dataproc Cloud Storage connector
https://cloud.google.com/dataproc/docs/concepts/connectors/cloud-storage

Customer use case: Best practices for migrating Hadoop to Dataproc by LiveRamp
https://cloud.google.com/blog/products/data-analytics/best-practices-for-migrating-hadoop-to-gcp-dataproc

Apache Hadoop is an open-source framework for big data. It is based on the MapReduce programming model, which Google invented and published. The MapReduce model, at its simplest, means that one function -- traditionally called the “map” function -- runs in parallel across a massive dataset to produce intermediate results; and another function -- traditionally called the “reduce” function -- builds a final result set based on all those intermediate results. The term “Hadoop” is often used informally to encompass Apache Hadoop itself and related projects, such as Apache Spark, Apache Pig, and Apache Hive.

Dataproc is a fast, easy, managed way to run Hadoop, Spark, Hive, and Pig on Google Cloud. All you have to do is to request a Hadoop cluster. It will be built for you

in 90 seconds or less, on top of Compute Engine virtual machines whose number and type you can control. If you need more or less processing power while your cluster's running, you can scale it up or down. You can use the default configuration for the Hadoop software in your cluster, or you can customize it. And you can monitor your cluster using Operations.

Hadoop history (simplified)

- Google and Yahoo were looking for ways to analyze mountains of user internet search results
  - Google published a white paper on MapReduce in 2004
  - Yahoo implemented the concepts and open sourced it in 2008
- Two main components when running on-premise
  - Multiple nodes (VMs) to process data
    - May consist of 1,000s of nodes
    - Shares computational workloads and works on data in parallel
  - Each node has persistent disk storage
    - Hadoop Distributed File System (HDFS) to store the data

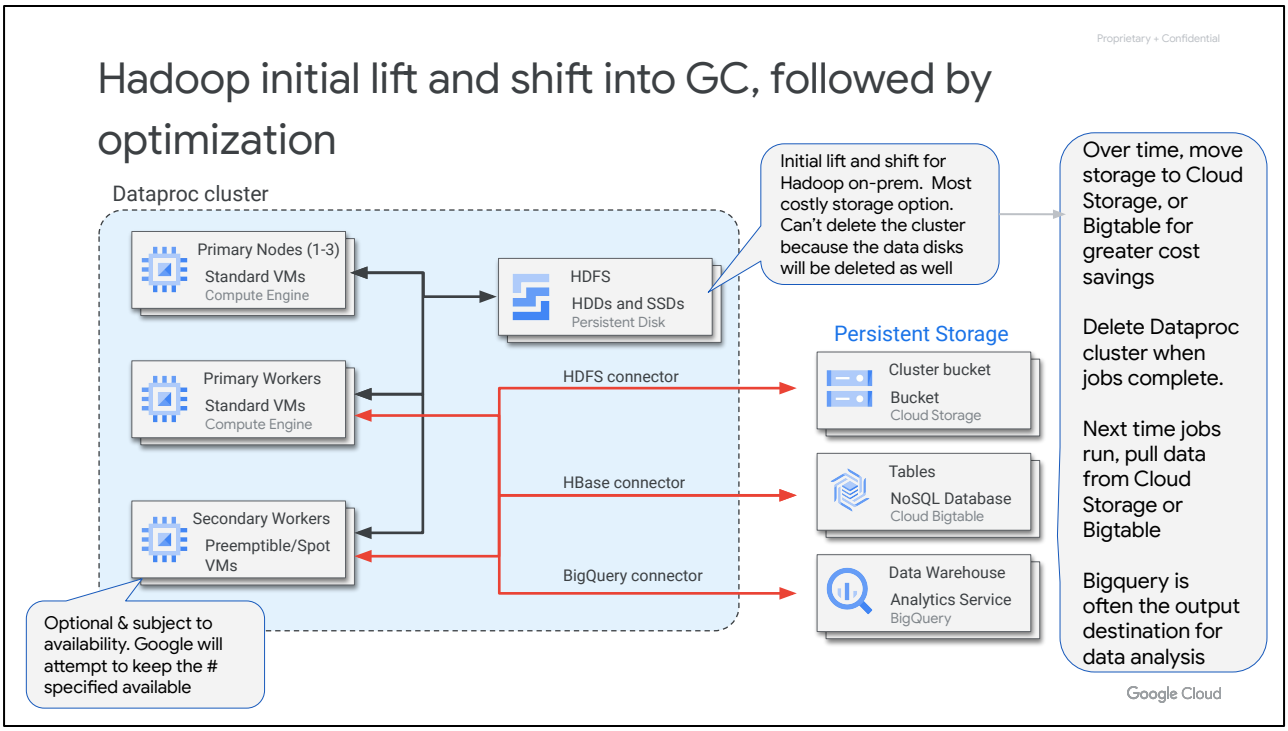
Google's whitepaper:
https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf

Yahoo released Hadoop as an open source project to Apache Software Foundation in 2008

Example use case - Clickstream data

- Websites often track every click made by every user on every page visited
  - Results in millions of rows of data
- Analysts would like to know why someone added items to a shopping cart, proceeded to checkout, and then abandoned the cart
  - Don't need *all* the data - just data for users who abandoned their cart
    - Phase 1 (aka "Map")
      - Process the data and get the users, the contents of their carts and the last page visited
    - Phase 2 (aka "Reduce")
      - Aggregate the total the number and value of carts abandoned per month
      - Plus total the most common final pages that someone viewed before ending the user session

This is a very simplistic example.



The lines in black are usually the initial implementation. Customers gain greater cost savings when they transition to the red flow. This requires making some modifications to the jobs to use the connectors.

HDFS: Hadoop data file system. Cloud storage was originally named DFS (distributed file system)
HBase: open-source, NoSQL, distributed big data store. Runs on top of HDFS. The GC equivalent is Bigtable

DFS - distributed file system = Cloud Storage

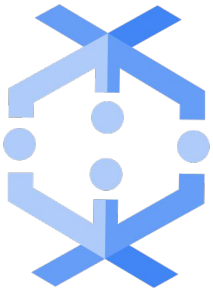
HA Dataproc has 3 masters (one is a witness); With no HA , have 1 master, no failover. All are in the same zone

Primary workers can use autoscaling. Secondary workers are MIGs but do not scale. However if you tell Google you want 4 workers, it will try to keep 4 workers at al times. These can be spot VMs

HBASE - database that lives in HDFS. Equivalent to Cloud Bigtable

Dataflow offers managed data pipelines

- Processes data using Compute Engine instances.
  - Clusters are sized for you
  - Automated scaling, no instance provisioning required
- Write code once and get batch and streaming
  - Transform-based programming model



Dataflow, the backbone of data analytics

Dataflow, the backbone of data analytics
https://cloud.google.com/blog/topics/developers-practitioners/dataflow-backbone-data-analytics

Dataflow Under the Hood: Comparing Dataflow with other tools (Aug 24, 2020)
https://cloud.google.com/blog/products/data-analytics/dataflow-vs-other-stream-batch-processing-engines

Dataproc is great when you have a dataset of known size, or when you want to manage your cluster size yourself. But what if your data shows up in realtime? Or it's of unpredictable size or rate? That's where Dataflow is a particularly good choice. It's both a unified programming model and a managed service, and it lets you develop and execute a big range of data processing patterns: extract-transform-and-load, batch computation, and continuous computation. You use Dataflow to build data pipelines, and the same pipelines work for both batch and streaming data.

Dataflow is a unified programming model and a managed service for developing and executing a wide range of data processing patterns including ETL, batch computation, and continuous computation. Dataflow frees you from operational tasks like resource management and performance optimization.

Dataflow features:

Resource Management: Dataflow fully automates management of required processing resources. No more spinning up instances by hand.

On Demand: All resources are provided on demand, enabling you to scale to meet your business needs. No need to buy reserved compute instances.

Intelligent Work Scheduling: Automated and optimized work partitioning which can dynamically rebalance lagging work. No more chasing down “hot keys” or pre-processing your input data.

Auto Scaling: Horizontal auto scaling of worker resources to meet optimum throughput requirements results in better overall price-to-performance.

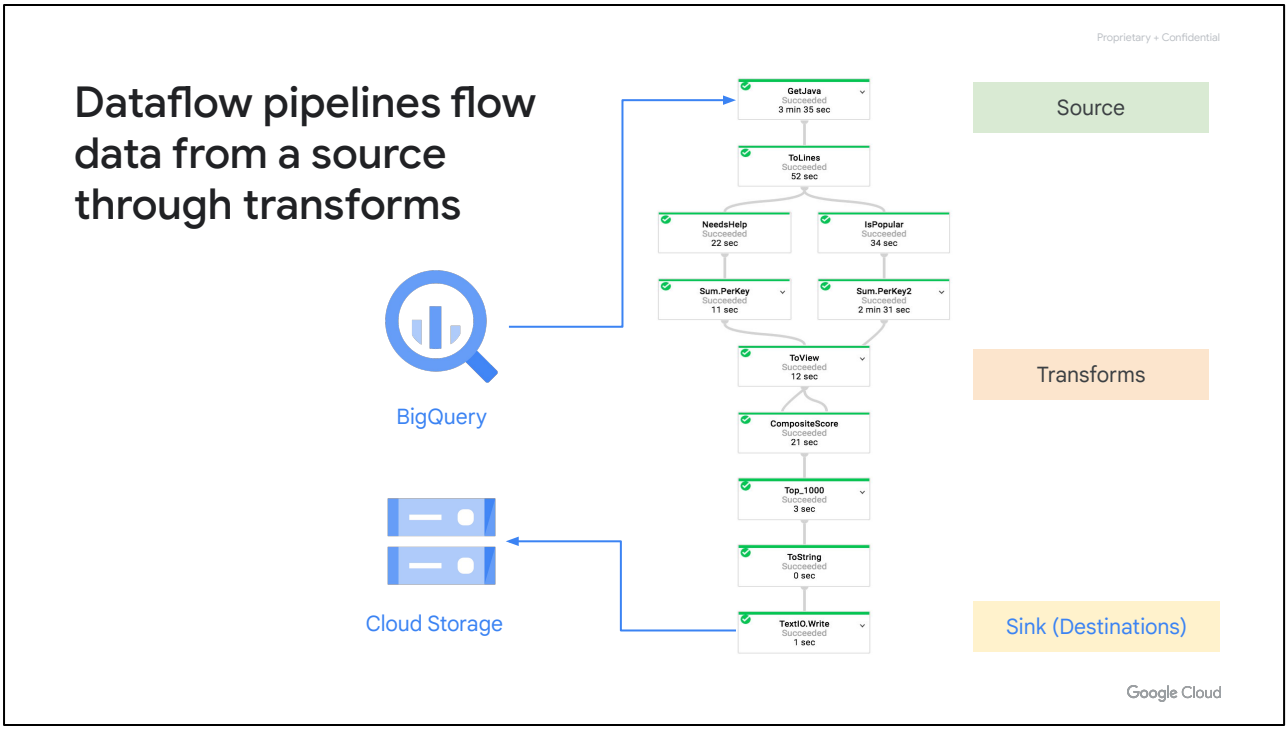
Unified Programming Model: The Dataflow API enables you to express MapReduce like operations, powerful data windowing, and fine grained correctness control regardless of data source.

Open Source: Developers wishing to extend the Dataflow programming model can fork and or submit pull requests on the Java-based Dataflow SDK. Dataflow pipelines can also run on alternate runtimes like Spark and Flink.

Monitoring: Integrated into the Cloud Console, Dataflow provides statistics such as pipeline throughput and lag, as well as consolidated worker log inspection—all in near-real time.

Integrated: Integrates with Cloud Storage, Pub/Sub, Datastore, Cloud Bigtable, and BigQuery for seamless data processing. And can be extended to interact with others sources and sinks like Apache Kafka and HDFS.

Reliable & Consistent Processing: Dataflow provides built-in support for fault-tolerant execution that is consistent and correct regardless of data size, cluster size, processing pattern or pipeline complexity.

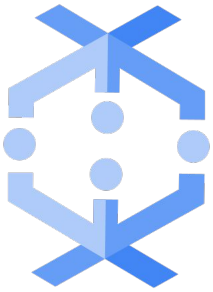


This example Dataflow pipeline reads data from a BigQuery table (the “source”), processes it in various ways (the “transforms”), and writes its output to Cloud Storage (the “sink”). Some of those transforms you see here are map operations, and some are reduce operations. You can build really expressive pipelines.

Each step in the pipeline is elastically scaled. There is no need to launch and manage a cluster. Instead, the service provides all resources on demand. It has automated and optimized work partitioning built in, which can dynamically rebalance lagging work. That reduces the need to worry about “hot keys” -- that is, situations where disproportionately large chunks of your input get mapped to the same custer.

Why use Dataflow?

- *ETL* (extract/transform/load) pipelines to move, filter, enrich, shape data
- *Data analysis*: batch computation or continuous computation using streaming
- *Orchestration*: create pipelines that coordinate services, including external services
- Integrates with Google Cloud services like Cloud Storage, Pub/Sub, BigQuery, and Cloud Bigtable
  - Open source Java, Python SDKs



People use Dataflow in a variety of use cases. For one, it serves well as a general-purpose ETL tool.

And its use case as a data analysis engine comes in handy in things like these: fraud detection in financial services; IoT analytics in manufacturing, healthcare, and logistics; and clickstream, Point-of-Sale, and segmentation analysis in retail.

And, because those pipelines we saw can orchestrate multiple services, even external services, it can be used in real time applications such as personalizing gaming user experiences.

Proprietary + Confidential

BigQuery is a fully managed data warehouse

Revisiting this topic

- Provides near real-time interactive analysis of massive datasets (hundreds of TBs).
- Query using SQL syntax (ANSI SQL 2011)
- No cluster maintenance is required
- Compute and storage are separated with a terabit network in between.
- You only pay for storage and processing used.
- Automatic discount for long-term data storage.

https://cloud.google.com/bigquery

Google Cloud

Query BIG with BigQuery: A cheat sheet
https://cloud.google.com/blog/topics/developers-practitioners/query-big-bigquery-cheat-sheet

If, instead of a dynamic pipeline, you want to do ad-hoc SQL queries on a massive dataset, that is what BigQuery is for. BigQuery is Google's fully managed, petabyte scale, low cost analytics data warehouse.

BigQuery is NoOps: there is no infrastructure to manage and you don't need a database administrator, so you can focus on analyzing data to find meaningful insights, use familiar SQL, and take advantage of our pay-as-you-go model. BigQuery is a powerful big data analytics platform used by all types of organizations, from startups to Fortune 500 companies.

BigQuery's features:

Flexible Data Ingestion: Load your data from Cloud Storage or Datastore, or stream it into BigQuery at 100,000 rows per second to enable real-time analysis of your data.

Global Availability: You have the option to store your BigQuery data in European locations while continuing to benefit from a fully managed service, now with the option of geographic data control, without low-level cluster maintenance.

Security and Permissions: You have full control over who has access to the data

stored in BigQuery. If you share datasets, doing so will not impact your cost or performance; those you share with pay for their own queries.

Cost Controls: BigQuery provides cost control mechanisms that enable you to cap your daily costs at an amount that you choose. For more information, see Cost Controls.

Highly Available: Transparent data replication in multiple geographies means that your data is available and durable even in the case of extreme failure modes.

Super Fast Performance: Run super-fast SQL queries against multiple terabytes of data in seconds, using the processing power of Google's infrastructure.

Fully Integrated In addition to SQL queries, you can easily read and write data in BigQuery via Dataflow, Spark, and Hadoop.

Connect with Google Products: You can automatically export your data from Google Analytics Premium into BigQuery and analyze datasets stored in Google Cloud Storage, Google Drive, and Google Sheets.

BigQuery can make Create, Replace, Update, and Delete changes to databases, subject to some limitations and with certain known issues.

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore , Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

BigQuery Data Transfer Service

- Automates data movement into BigQuery on a scheduled, managed basis
- Transfers data into to BigQuery (does not transfer data out of BigQuery)
- Supports loading data from the following data sources*
  - External cloud storage providers
    - Amazon S3
  - Data warehouses
    - Teradata
    - Amazon Redshift
  - Google Software as a Service (SaaS) apps
  - Campaign Manager
  - Cloud Storage
  - Google Ad Manager

*Check the documentation for a current list of data sources

BigQuery Data Transfer Service
https://cloud.google.com/bigquery-transfer/docs/introduction

BigQuery - other data ingestion methods

- In addition to the Data Transfer Service, are several others ways to ingest data into BigQuery:
  - Batch load a set of data records
    - Sources could be Cloud Storage, or file stored on a local machine
    - Data could be formatted in Avro, CSV, JSON, ORC, or Parquet
  - Stream individual records or batches of records
    - A Storage Write API for BigQuery introduced in 2021
      - Data must be in the ProtoBuf format
      - This is a detailed coding solution that provides 100% control
      - Alternatively, use Pub/Sub and DataFlow
        - Much of the complexity is handled by Google, e.g., autoscaling
  - Use SQL queries to generate new data and append or overwrite the results to a table.
  - Use a third-party application or service

Introduction to loading data
https://cloud.google.com/bigquery/docs/loading-data

Cloud Storage - choosing a transfer option

Where you're moving data from	Scenario	Suggested products
Another cloud provider (for example, Amazon Web Services or Microsoft Azure) to Google Cloud	—	Storage Transfer Service
Cloud Storage to Cloud Storage (two different buckets)	—	Storage Transfer Service
Your private data center to Google Cloud	Enough bandwidth to meet your project deadline for less than 1 TB of data	gsutil
Your private data center to Google Cloud	Enough bandwidth to meet your project deadline for more than 1 TB of data	Storage Transfer Service for on-premises data
Your private data center to Google Cloud	Not enough bandwidth to meet your project deadline	Transfer Appliance

Cloud SQL

- Export data to a storage bucket

```
gcloud sql export csv my-cloudsqlserver
gs://my-bucket/sql-export.csv \
--database=hr-database \
--query='select * from employees'
```

- Import data from a bucket

```
gcloud sql import sql [INSTANCE_NAME]
gs://[BUCKET_NAME]/[IMPORT_FILE_NAME] \
--database=[DATABASE_NAME]
```

- For REST API examples, see https://cloud.google.com/sql/docs/mysql/import-export/import-export-csv#export_data_to_a_csv_file

Cloud SQL - Export and import using CSV files:
https://cloud.google.com/sql/docs/mysql/import-export/import-export-csv

Cloud SQL - Best practices for importing and exporting data:
https://cloud.google.com/sql/docs/mysql/import-export/

Streaming data to Pub/Sub

- Suggest looking at various quickstarts
  - Stream messages from Pub/Sub by using Dataflow
    - https://cloud.google.com/pubsub/docs/stream-messages-dataflow
  - Using client libraries
    - https://cloud.google.com/pubsub/docs/publish-receive-messages-client-library
  - Using gcloud CLI
    - https://cloud.google.com/pubsub/docs/publish-receive-messages-gcloud

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

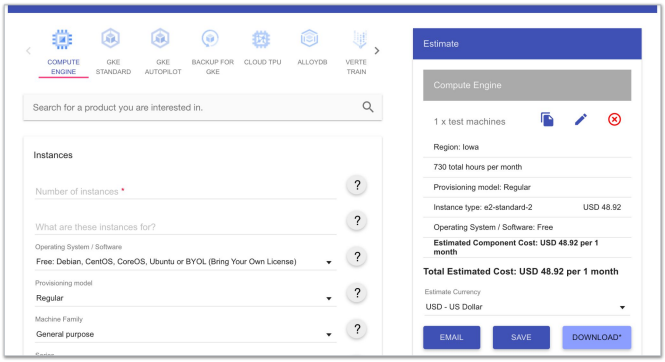
- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore , Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Use the Google Cloud Pricing Calculator to estimate costs

- Create cost estimates based on forecasting and capacity planning.
- The parameters entered will vary according to the service, e.g.,
  - Compute Engine - machine type, operating system, usage/day, disk size, etc
  - Cloud Storage - Location, storage class, storage amount, ingress and egress estimates
- Can save and email estimates for later use, e.g., presentations



https://cloud.google.com/products/calculator

The pricing calculator is the go-to resource for gaining cost estimates. Remember that the costs are just an estimate, and actual cost may be higher or lower. The estimates by default use the timeframe of one month. If any inputs vary from this, they will state this. For example, Firestore document operations read, write, and delete are asked for on a per day basis.

BigQuery: Executing queries in the CLI with `dry_run`

```
-bq query --use_legacy_sql=false --dry_run \
'SELECT
  word,
  SUM(word_count) AS count
FROM
`bigquery-public-data`.samples.shakespeare
WHERE
  word LIKE "%raisin%"
GROUP BY
  word'
```

Add `--dry_run` flag to receive estimate of how much data will be processed

Plug that number into the Pricing Calculator

BigQuery - Estimate storage and query costs:
https://cloud.google.com/bigquery/docs/estimate-costs

BigQuery - estimate query costs:
https://cloud.google.com/bigquery/docs/estimate-costs#estimate_query_costs

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

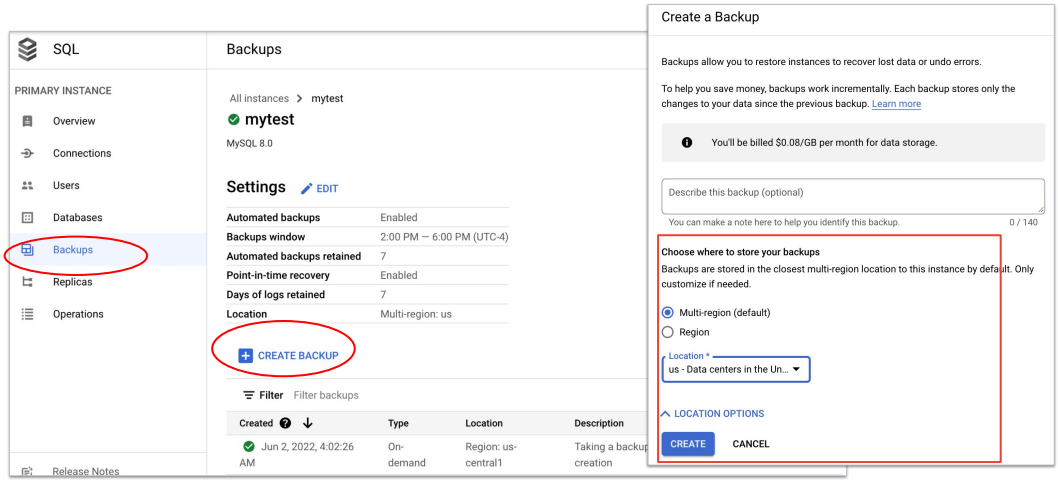
4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore , Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Cloud Spanner backup and restore:
https://cloud.google.com/spanner/docs/backup

Bigtable - Manage backups:
https://cloud.google.com/bigtable/docs/managing-backups

Cloud SQL Backup - Console

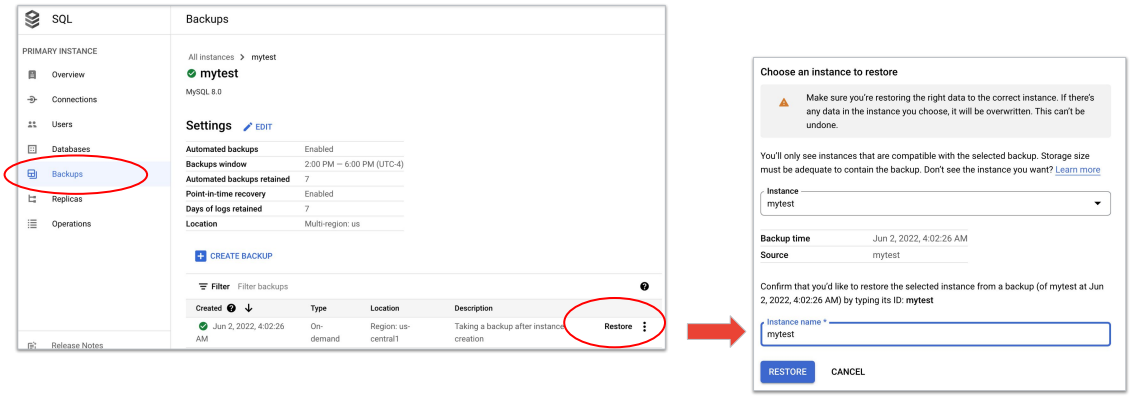


About Cloud SQL backups:
https://cloud.google.com/sql/docs/mysql/backup-recovery/backups

Cloud SQL - Create and manage on-demand and automatic backups:
https://cloud.google.com/sql/docs/mysql/backup-recovery/backing-up

Cloud SQL - Schedule Cloud SQL database backups:
https://cloud.google.com/sql/docs/mysql/backup-recovery/scheduling-backups

Cloud SQL Restore - Console



Cloud SQL Backup/Restore - CLI

`--async` means don't wait for the command to complete

- Create backup

```
gcloud sql backups create --async --instance mytest
```

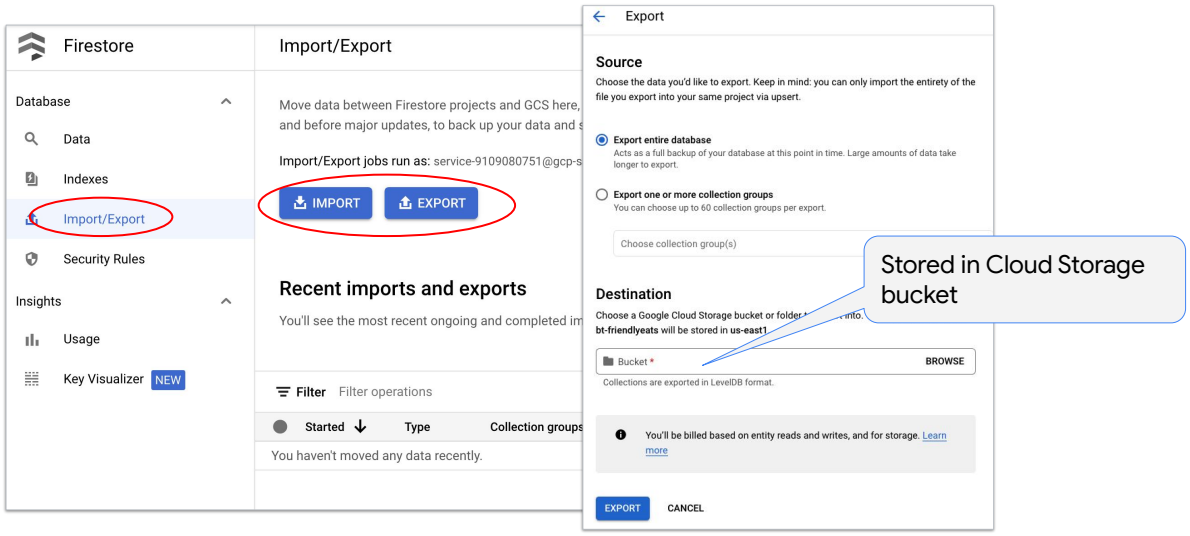
- Backups - get the instance IDs

```
gcloud sql backups list --instance mytest
```

- Restore backup - get the backup ID

```
gcloud sql backups restore [Backup ID] \
--restore-instance mytest \
--backup-instance mytest \
--async
```

Cloud Firestore Backup/Restore - Console



Firestore (Datastore) - Exporting and Importing Entities:
https://cloud.google.com/datastore/docs/export-import-entities

Firestore (Datastore) - Move data between projects:
https://firebase.google.com/docs/firestore/manage-data/move-data

Cloud Firestore Export/Import - CLI

- Export data

```
gcloud firestore export gs://my-firestore-data
```

- Import data

```
gcloud firestore import
gs://my-firestore-data/file-created-by-export/
```

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore , Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery

Dataproc: Reviewing job status - CLI

- Getting job status

All jobs

```
gcloud dataproc jobs list
```

Specific job

```
gcloud dataproc jobs describe job-id
--region=region
```


What is Dataproc?:
https://cloud.google.com/dataproc/docs/concepts/overview

Life of a Dataproc Job:
https://cloud.google.com/dataproc/docs/concepts/jobs/life-of-a-job

gcloud command:
https://cloud.google.com/sdk/gcloud/reference/dataproc/jobs/list

Dataflow: Reviewing job status - CLI

- Getting job status

All jobs

```
gcloud dataflow jobs list
```

Specific job

```
gcloud dataflow jobs describe job-id
--region=region
```


Using the Dataflow monitoring interface:
https://cloud.google.com/dataflow/docs/guides/using-monitoring-intf

gcloud command:
https://cloud.google.com/sdk/gcloud/reference/dataflow/jobs/describe

BigQuery: Reviewing job status - CLI

- Listing jobs

```
bq ls --jobs=true --all=true
```

- Getting job status

```
bq --location=LOCATION show --job=true JOB_ID
```

- Example

```
bq show --job=true
myproject:US.bquijob_123x456_123y123z123c
```

- Sample output

```
Job Type    State      Start Time      Duration      User Email       Bytes Processed   Bytes Billed
 ---------- --------- ----------------- ---------- ------------------ ----------------- --------------
    extract    SUCCESS   06 Jul 11:32:10   0:01:41    user@example.com
```


BigQuery - Introduction to BigQuery jobs:
https://cloud.google.com/bigquery/docs/jobs-overview

BigQuery - Managing jobs:
https://cloud.google.com/bigquery/docs/managing-jobs

View job details with bg cli:
https://cloud.google.com/bigquery/docs/managing-jobs#bq

Exam Guide - Storage options

2.3 Planning and configuring data storage options. Considerations include:

- 2.3.1 Product choice (e.g., Cloud SQL, BigQuery, Firestore, Cloud Spanner, Cloud Bigtable)
- 2.3.2 Choosing storage options (e.g., Zonal persistent disk, Regional balanced persistent disk, Standard, Nearline, Coldline, Archive)

3.4 Deploying and implementing data solutions. Tasks include:

- 3.4.1 Initializing data systems with products (e.g., Cloud SQL, Firestore, BigQuery, Cloud Spanner, Pub/Sub, Cloud Bigtable, Dataproc, Dataflow, Cloud Storage)
- 3.4.2 Loading data (e.g., command line upload, API transfer, import/export, load data from Cloud Storage, streaming data to Pub/Sub)

4.4 Managing storage and database solutions. Tasks include:

- 4.4.1 Managing and securing objects in and between Cloud Storage buckets
- 4.4.2 Setting object life cycle management policies for Cloud Storage buckets
- 4.4.3 Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Datastore , Cloud Bigtable)
- 4.4.4 Estimating costs of data storage resources
- 4.4.5 Backing up and restoring database instances (e.g., Cloud SQL, Datastore)
- 4.4.6 Reviewing job status in Dataproc, Dataflow, or BigQuery


Google Cloud