



# Google Cloud Data Analytics

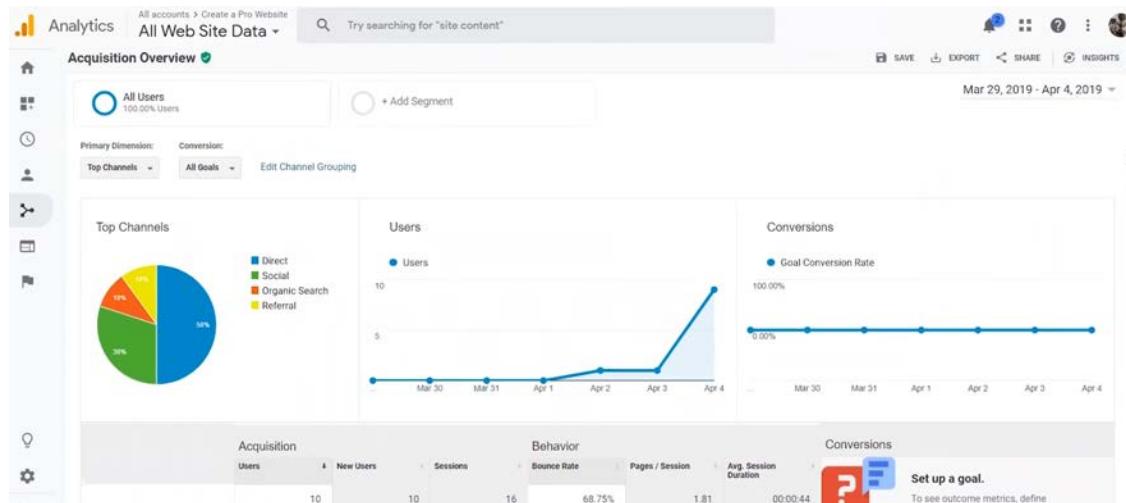
## ▼ Document 1

### ▼ Google Analytics

#### ▼ What is Google Analytics?

Google Analytics is an essential tool for tracking and analyzing website traffic, helping website owners and digital marketers understand how users interact with their content. This document explains what Google Analytics is, how it works, and why it's important for website growth.

#### 1. Introduction to Google Analytics

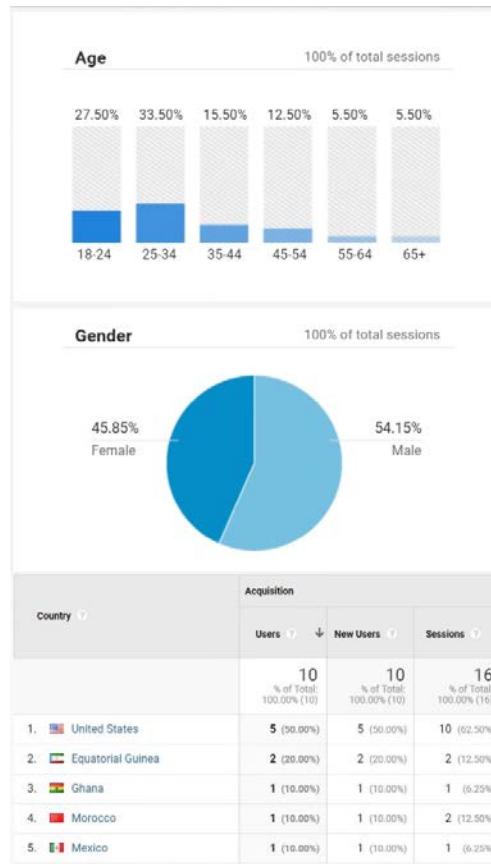


- Google Analytics is a powerful plugin that provides detailed insights into your website's performance.

- It tracks website traffic, which refers to the number of visitors coming to your site.
- This tool is especially valuable for anyone aiming to grow their website through digital marketing efforts, helping them make data-driven decisions.

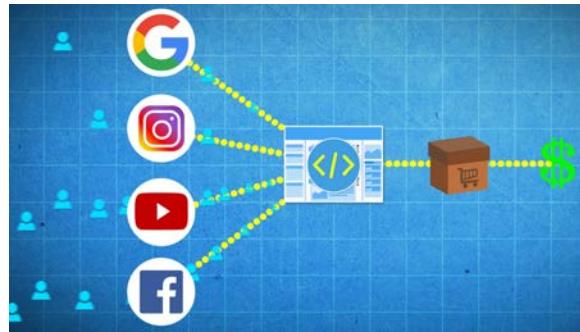
## 2. What Can Google Analytics Track?

- **Visitor Data:** Google Analytics tracks how many visitors your site receives and where they are coming from (e.g., Google, YouTube, Facebook).
- **User Behavior:** It shows how long visitors stay on your site, which pages they view, and what they click on.
- **Demographics:** It can provide information about the age, gender, location (city or country), and device type of your visitors.



- Without Google Analytics, managing a website is like "driving a cruise ship with a blindfold," making it difficult to improve your site effectively.

## 3. How Google Analytics Works



- After setting up Google Analytics, a special tracking code is inserted into your website. This code gathers data on your visitors' activities.
- No coding knowledge is required to install the tracking code. There are simple tutorials available to guide you through the process, including how to set it up on platforms like WordPress.
- Once the code is in place, Google Analytics will:
  - Track where visitors are coming from (e.g., which ad or platform is driving the most traffic).
  - Provide insights on the performance of your campaigns, such as comparing video ads to picture ads.
  - Help you understand which parts of your website need improvement, such as if visitors are leaving from your homepage, product page, or checkout cart (commonly known as **bounce rate**).

## 4. Optimizing Website Performance

- By using the data from Google Analytics, you can improve your website's user experience and optimize your marketing efforts.
- You'll be able to identify where visitors are dropping off and take steps to fix any issues.
- You can also see demographic information about your audience, allowing you to target the best-performing segments and grow your website further.

## 5. Conclusion

- Google Analytics is crucial for anyone who wants to understand and improve their website's performance.
- It provides detailed insights into visitor behavior, helping you identify what works and what doesn't.

- Whether you're running ads, launching new content, or selling products online, Google Analytics is the tool that can guide your website toward success.

For more information and tutorials on setting up websites and leveraging Google Analytics, check out relevant resources or consider subscribing to content that offers in-depth guidance.

## ▼ Google Analytics vs. Data Analysis

In the world of digital marketing and business intelligence, both Google Analytics and data analysis play vital roles. However, they serve different purposes and operate at different levels of detail. This document compares Google Analytics with broader data analysis practices to highlight their key differences and uses.

### 1. What is Google Analytics?

- **Google Analytics** is a specialized web analytics tool that focuses on tracking and reporting website traffic. It provides insights into how users interact with a website, offering data on:
  - Website visitors (traffic volume, sources)
  - User behavior (bounce rates, time spent on pages)
  - Demographics (age, gender, location)
  - Device usage (mobile, desktop)
  - Conversion rates (purchases, sign-ups)
- **Purpose:** It is specifically designed to help website owners and marketers understand online user behavior and optimize website performance for improved results.

### 2. What is Data Analysis?

- **Data Analysis** is a broader process that involves inspecting, cleaning, transforming, and modeling data to discover useful information, draw conclusions, and support decision-making. It is used across various industries (not limited to websites) and includes:
  - Data mining
  - Statistical analysis
  - Machine learning
  - Predictive modeling
  - AI

- **Purpose:** The goal of data analysis is to extract actionable insights from raw data, regardless of the source (e.g., sales data, financial data, customer data).

### 3. Key Differences

| Aspect                | Google Analytics                                | Data Analysis  |
|-----------------------|---|--|
| <b>Scope</b>          | Website-specific data                           | Any kind of data (e.g., sales, customer, operational)                                      |
| <b>Primary Use</b>    | Web traffic monitoring, digital marketing       | Business intelligence, decision-making in various fields                                   |
| <b>Data Sources</b>   | Website, app, online platforms                  | Any structured or unstructured data sources  |
| <b>Ease of Use</b>    | User-friendly dashboard with predefined metrics | Requires data processing and possibly coding knowledge (e.g., SQL, Python, R)              |
| <b>Customization</b>  | Limited to pre-defined website data points      | Highly customizable for various use cases  |
| <b>Real-Time Data</b> | Real-time web traffic reports                   | Data analysis can include both real-time and historical data, depending on tools and needs |
| <b>Reporting</b>      | Prebuilt reports, easy-to-use interface         | Often requires custom queries, scripting, and report building                              |
| <b>Tools Used</b>     | Google Analytics, Tag Manager                   | SQL, Excel, Python, R, Tableau, Power BI, etc.   |

### 4. When to Use Google Analytics?

- **Web Traffic Analysis:** If you need to know how visitors are interacting with your website, where they come from, and how well your site is converting, Google Analytics is your go-to tool.
- **Campaign Tracking:** Google Analytics is ideal for monitoring the success of digital marketing campaigns like Google Ads, YouTube ads, or social media campaigns.
- **User Behavior:** It's perfect for understanding on-site behavior like time spent on pages, popular content, and user navigation paths.

### 5. When to Use Data Analysis?

- **Business-Wide Analysis:** For data that spans beyond website traffic, like analyzing sales trends, customer demographics, or operational efficiency, you need data analysis techniques.
- **Custom Insights:** If your questions are not answered by standard Google Analytics reports (e.g., predictive modeling, advanced segmentation), data analysis provides the flexibility and tools to dive deeper.

- **Multiple Data Sources:** When you're dealing with data from multiple sources (e.g., CRM, financial systems, supply chain), data analysis techniques are necessary to combine and analyze this data.

## 6. How They Work Together

- **Complementary Tools:** Google Analytics can be seen as a data source for a broader data analysis process. For example, you may export data from Google Analytics into a data analysis tool like Python or Power BI to perform deeper analytics.
- **Integration:** Google Analytics data can be combined with other data sources (e.g., sales, customer data) to build a complete view of your business performance, driving more informed decision-making.

## 7. Conclusion

- **Google Analytics** is a web-specific tool designed to provide insights into user behavior on websites and apps, ideal for digital marketers and website owners.
- **Data Analysis** is a broader discipline that applies to various fields and requires deeper technical skills to analyze data from different sources for business intelligence and decision-making.
- Both tools are essential, but they serve different needs: use **Google Analytics** for web data and **data analysis** for comprehensive, multi-source insights.

Both Google Analytics and data analysis are integral to building a data-driven strategy, but their use depends on the scale and nature of your data needs.

# ▼ 1- Introduction to Data Analytics in Google Cloud

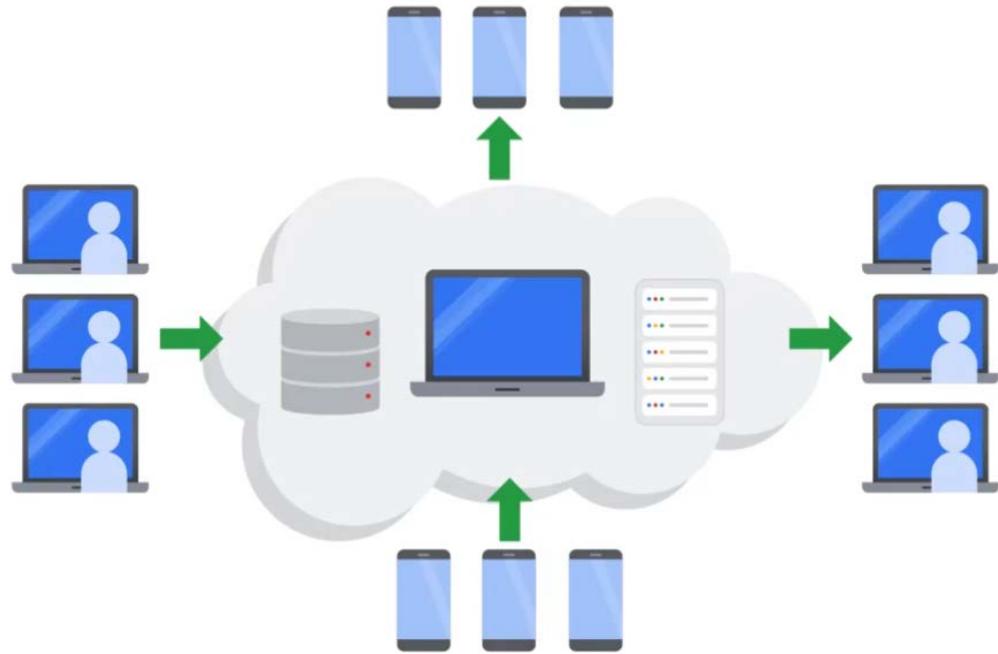
## ▼ Introduction to Cloud Computing

### ▼ Basics of Cloud Computing

What exactly is cloud computing?

- **Cloud computing** is the practice of using on-demand computing resources as services hosted over the internet.
- “Over the internet” is what makes up the cloud part. It eliminates the need for organizations to find, set up, or manage resources themselves.
- Cloud computing uses a network to connect users to a cloud platform.
- This is a virtual space where they can access and borrow computing services.

# Cloud computing network



A primary computer handles all communication between devices and servers to share information. And there are privacy and security measures to keep everything safe. The cloud enables organizations to access computing resources on-demand, without spending time and money buying and maintaining their own storage, hardware, and software. It's the unique infrastructure of a cloud computing model that makes all of this possible.

This infrastructure has four main components:

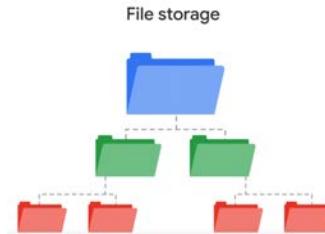
- hardware
- storage,
- network,
- virtualization.

1. **Hardware:** Types of hardware include servers, processors, and memory; network switches, routers, and cables; firewalls and load balancers; cooling systems; and power supplies.

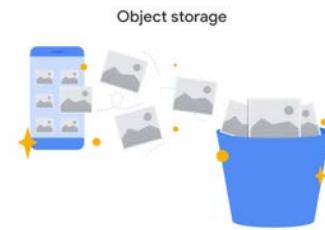
2. **Storage:**

Data storage in a cloud computing infrastructure can occur in three main ways: file, object, or block.

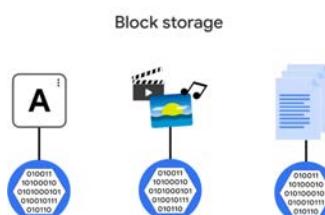
1. File storage keeps data in one place and organizes it in a simple, easy-to-understand way through a hierarchy of files in folders.



2. Object storage holds unstructured data, along with its metadata. Metadata is just data about data. For example, a picture taken with a smartphone might contain information about the location, the date, and the type of device that captured the image.



3. Block storage divides large volumes of data into smaller pieces, optimally organized, with unique labels. An advantage of block storage is that the data is easily accessible, but it can be expensive and has limited capability to handle metadata.



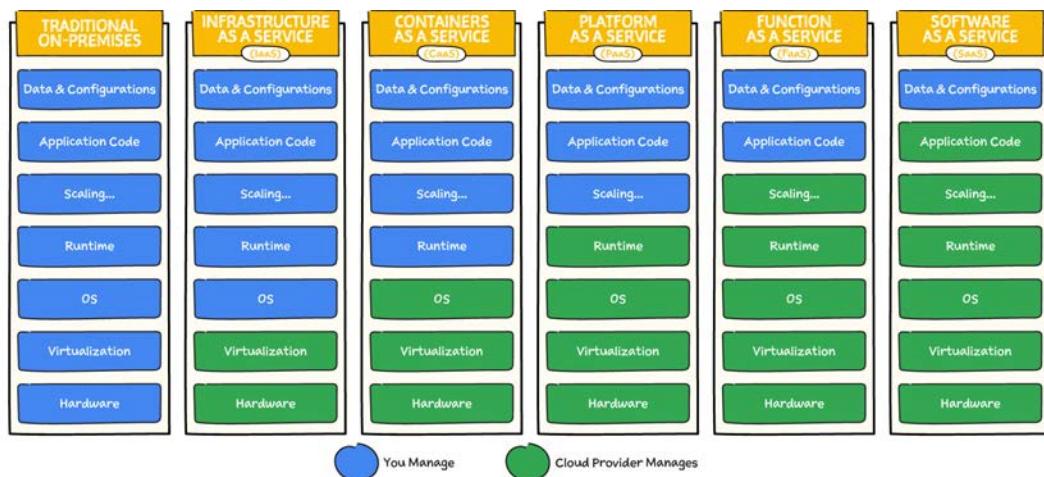
3. **Network:** Cloud computing infrastructure needs a way to connect to its backend resources, and this connection is made possible through a network of the physical hardware. Through this network, users tap into cloud resources using some of the hardware mentioned earlier, including routers and firewalls.
4. **Virtualization:** Basically, the physical network setup is what enables the virtual one to operate. Finally, there is virtualization, which is technology that creates a virtual version of physical infrastructure, like servers, storage, and networks. This is what lets the service work without a connection.

## ▼ Cloud Computing Service Models

A data center is a physical building that contains servers, computer systems, and associated components. These facilities provide a centralized location for vast amounts of data.

For all sorts of business tasks and projects, cloud analysts select and extract relevant data, then prepare it for processing and examination. They know how to expertly analyze, visualize, and share data discoveries to uncover valuable insights and make smart business decisions. So it's really important to know that there are three primary models to choose from. Each offers a different level of flexibility and control.

The cloud computing model is the structure that determines how cloud services will be offered and used. We can define cloud computing as a service that delivers resources (servers, storage, network, software, etc.) on demand over the internet.



These models are:

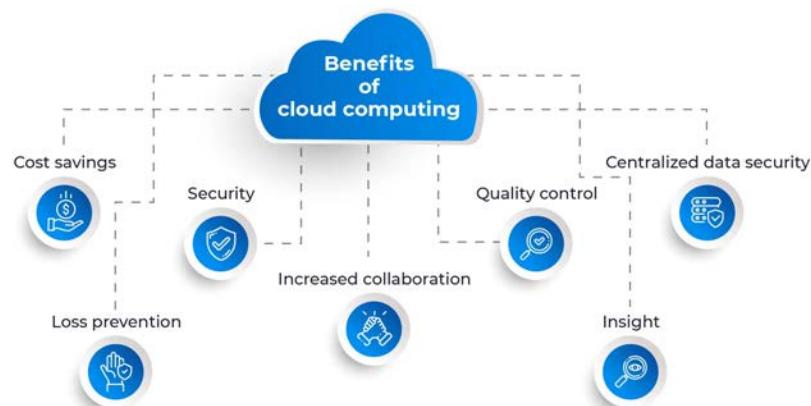
- Infrastructure as a Service, IaaS
- Platform as a Service, PaaS
- Software as a Service, SaaS

1. First, IaaS is a cloud computing module that offers on-demand access to Information Technology, or IT infrastructure services, including hardware, storage, network, and virtualization tools. With IaaS, a service provider hosts, maintains, and updates the infrastructure. Your organization would manage everything else: your operating system, your data, and your applications. An IaaS model provides the highest level of control over your IT resources and

works a lot like traditional on-premises IT. An example of an IaaS model is cloud storage, like emails you've sorted into an online folder. Here's another example. When someone leases a car, it's like they're borrowing it for a while, having fun driving it around, but they have to give it back when the lease agreement is up. Well, IaaS is kind of like that. A user picks the infrastructure they want, uses it for the contracted period, but they do not own it.

2. PaaS provides hardware and software tools to create an environment for the development of cloud applications, simplifying the application-development process. PaaS is all about helping users build apps. So, your organization would enjoy being able to fully focus on app development without the burden of managing and maintaining the underlying infrastructure. Your developers would create, test, troubleshoot, launch, host, and maintain your app all on the platform. PaaS is like hopping into a taxi and telling the driver where to take you. You're not behind the wheel, but you trust the driver to get you to where you need to be.
3. SaaS provides users with a licensed subscription to a complete software package. This includes the infrastructure, maintenance and updates, and the application itself. Other users also have access to use the same services. Using SaaS, you'd just connect to the app through the internet. Think of SaaS like riding the bus —you pick your stop from routes that are set already and you share the bus ride with other people. Remember that these examples are meant to demonstrate the level of individual customization in IaaS, PaaS, and SaaS. They do not refer to any actual hardware or software details.

## ▼ Benefits of Cloud Computing



1. **Accessibility:** Access and manage data, software, storage and cloud infrastructure

- One of the big advantages of a cloud computing model is that organizations can access and manage data, software, storage, and cloud infrastructure from any location at any time through the internet. They don't need to be physically present where the hardware and software are installed. And they don't need their cloud service provider's assistance when they need more data.

2. **Scalability:** Easily expand or upgrade computing resources to meet changing needs

- Scalability, which means to easily expand or upgrade computing resources to meet changing needs. Scalability eliminates physical computing limitations.

3. **Cost savings:** Only pay for the computing resources used.

- Benefit of cost savings is pretty straightforward. Organizations only pay for the computing resources used. In a cloud computing model, organizations get what's called a **measured service**. Similar to household utilities like electricity and water, users are charged only for what they use, based on the number of transactions, the storage volume, and the amount of data transferred. This helps make all kinds of business initiatives more profitable and sustainable.

4. **Security:** Systems, data and computing resources are protected from theft, damage, loss and unauthorized use

- The advantage of security is also pretty straightforward. With cloud computing, an organization's systems, data, and computing resources are protected from theft, damage, loss, and unauthorized use. Cloud computing security is generally recognized as stronger than the security of a traditional network infrastructure. This is because data is located in data centers that few people have access to. Plus, the information stored on cloud servers is encrypted, meaning it's not something that's easily broken into.

5. **Efficiency:** Immediate access to new and upgraded applications

- There's a lot that's efficient about cloud computing, but one of the main advantages is that organizations can provide immediate access to new and upgraded applications. There's no time wasted worrying about the state of network infrastructure or going through a costly or time-consuming implementation process.

6. **Managed Services**

A further benefit is the freeing up of internal resources. With managed services, third-party providers handle the maintenance, management, and support of cloud infrastructure and applications, allowing users to focus on higher-value tasks. It's like a mechanic who comes to you for annual services, rather than you spending time in the shop. All the background maintenance occurs automatically.

### Versatile Applications

Cloud computing's versatility opens up a wide range of uses, including:

1. **Disaster Recovery:** Cloud-based disaster recovery provides access to multiple data centers to safeguard data during emergencies.
2. **Data Storage:** Cloud data storage streamlines access, analysis, and backup of large data volumes, improving data center efficiency.
3. **Large-Scale Data Analysis:** Cloud computing offers quick access to various data sources and user-friendly interfaces for querying and exploring data, speeding up data-driven insights.

## ▼ Traditional vs Cloud Computing



### What is Traditional Computing?

Traditional computing is a model that enables data storage, access, and management through physical hardware and software within a network infrastructure, typically located on-premises. Here's how it works:

1. **Hardware Setup:** IT professionals set up hardware in a dedicated space or room.

2. **Software Installation:** Operating systems, applications, and security tools are purchased and installed.
3. **Maintenance:** Once operational, IT personnel maintain and manage the system.

This infrastructure gives an organization sole control and access to its data and equipment. Everything is located in one place and can't be accessed anywhere else.

## Advantages of Traditional Computing

Traditional computing offers four key advantages:

1. **Control:** Organizations have full control over hardware, software, and data. They can customize their localized network infrastructure to meet specific needs.
2. **Security:** Sole access to systems and sensitive information, if properly maintained, enhances security.
3. **Compliance:** Some industries require data to be stored on-premises for regulatory compliance, making traditional computing a necessity.
4. **No Reliance on the Internet:** Users can access the network and data even without an internet connection.

## Disadvantages of Traditional Computing

Just like with the photo album example, there are downsides to traditional computing:

1. **Limited Data Access:** Access is restricted to the location where hardware and software are installed.
2. **Scaling Challenges:** Scaling up requires more software, hardware, time, and physical space, making it expensive and difficult.
3. **Cost:** Traditional computing involves purchasing hardware, software, and ongoing maintenance, all of which add to the cost.

For these reasons, many organizations are moving to the cloud. The cloud is more accessible, scalable, and cost-effective. It's secure, efficient, and frees up staff to focus on more projects.

## ▼ Cloud Data Warehouses

Cloud data warehouse is a large-scale, data storage solution hosted on remote servers by a cloud service provider. To understand this better, picture it like a huge warehouse where large amounts of different types of containers from various places are stored.

The cloud data warehouse can collect, store, integrate, and analyze data.

## Advantages of Data Warehouses

- Fully managed by the cloud provider.
  - Cloud data warehouses are typically fully managed by the cloud provider.
  - This means that the cloud provider takes care of various operational tasks and maintenance, allowing users to focus on utilizing the data and gathering insights rather than handling the underlying infrastructure.
- Saves time, money and resources.
  - This saves time, money, and resources.
- More uptime compared to on-premises data warehouses.
  - Cloud data warehouses also have more uptime compared to on-premises data warehouses.

Uptime is the amount of time a machine is operational.

- Ability to scale and support increased demand for data
  - And, of course, only working computers have the ability to scale and support increased demands for data.
- Integration of separated data
  - Cloud warehouses can integrate separated data by gathering data from various structured sources within an organization, like sales systems, email lists, and websites, and pulling it all into one place.
  - This integrated data then can be analyzed for some pretty exciting and useful business insights.
- Real-time analytics
  - Another big advantage is that cloud data warehouses provide real-time analytics, ensuring users have quick access to the latest information. And in business, being fast is usually the key to outperforming the competition.
- AI and ML

- Cloud data warehouses also offer some really cool artificial intelligence, or AI, and machine learning, or ML, capabilities.
- And when you apply AI and ML to your data analysis, this really powers up the possibilities.
- Custom reporting and analysis
  - Cloud data warehouses enable custom reporting and analysis.
  - This means that users can analyze and generate reports specifically from historical data because it is stored on a separate server from data related to current business transactions and day-to-day operations.

## ▼ Introduction to BigQuery

BigQuery: Google's powerhouse of storage and analysis.

An organization's data is vital to its business success. And data warehousing helps make the most of that data by providing quick and easy access to information, which leads to ideas, insights, and, best of all, data-driven decision-making.

- BigQuery is a data warehouse on Google Cloud that helps users store and analyze data right within BigQuery.
- They can query data, filter large datasets, aggregate results, and perform some really complex operations.
- BigQuery works with SQL, or structured query language. This is a computer programming language used to communicate with the database.
- It allows users to search through massive amounts of data—and find information they are searching for— incredibly quickly using Google infrastructure.
- Another feature is BigQuery's ability to easily migrate existing data warehouses from other cloud service providers.

## ▼ Data Analytics in the Cloud

### ▼ Steps for Effective Cloud Migration

We'll discuss the process of migrating an on-premises computing network infrastructure to a cloud platform.

- All right, the first step is to think about some key factors. These include choosing the right cloud environment for your organization.

## Migration to the Cloud



- Then, think about how much data will be transferred to the cloud. This is important because large amounts of data can take a long time to move, which can delay operations.
- Next, consider how much downtime your organization can deal with during migration. Obviously, no business wants to shut down their systems any longer than necessary, so this decision should be agreed on by all stakeholders.
- The next step is to choose your migration strategy. Options include rehosting, also called lift and shift, replatforming, repurchasing, refactoring, or retiring.

### Rehosting or “lift and shift”

A cloud-migration strategy that involves moving an entire on-premises system to the cloud.

- Rehosting is a cloud-migration strategy that involves moving an entire on-premises system to the cloud without changing anything else about the system.
- An exact copy of the current setup is created in the cloud, which helps the organization quickly achieve a return on investment as they use the enhanced efficiency of their operations, the robust and reliable nature of the cloud infrastructure, and the innovative technologies that are built into cloud-based solutions.

### Replatforming

A cloud-migration strategy that involves making small changes to the on-premises system once it's migrated to the cloud

- So, the main structure of the system's applications remains the same, but a few things are improved for better performance.

### Repurchasing

A cloud-migration strategy that involves moving applications to

a new, cloud-based service platform, usually a software-as-a-service platform.

- The cloud service will be an all-new experience, so this requires some team member training.

### Refactoring

A cloud-migration strategy that involves building all-new applications from scratch and discarding old applications.

- This is ideal when organizations need new features, like serverless computing, that their current systems don't have.

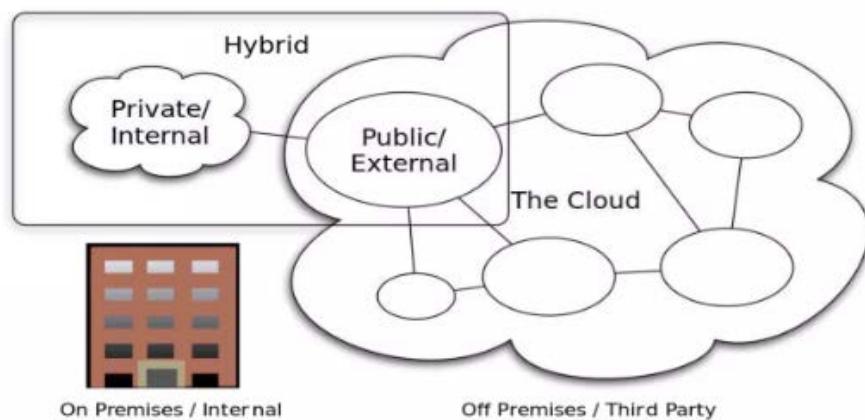
### Retiring

Applications that are no longer useful are turned off.

#### Migration path phases

- Assess
- Plan
- Deploy
- Optimize

## ▼ Explore Cloud Deployment Models



#### Cloud deployment models:

- **Public Clouds**

- First up, a public cloud is a cloud model that delivers computing, storage, and network resources through the internet. In this model, these resources are shared among multiple users and organizations, granting them on-demand access and utilization. Public cloud services are overseen and maintained by third-party cloud service providers, who not only manage the infrastructure but also operate their own data centers.

- **Private Clouds**

- Next, a private cloud is a cloud model that dedicates all cloud resources to a single user or organization and is created, managed, and owned within on-premises data centers.

- **Hybrid Clouds**

- Finally, hybrid clouds are a combination of the public and private models. They enable organizations to enjoy both cloud services and the control features of on-premises cloud models.

## **Advantages of Public Cloud**

- Pay only for what you use
- Scale up or down
- No maintenance
- Reliability
- Speed and ease of deployment
- New services

## **Advantages of Private Cloud**

- Private and secure networks
- Compliance
- Consistent Performance

## **Advantages of Hybrid Cloud**

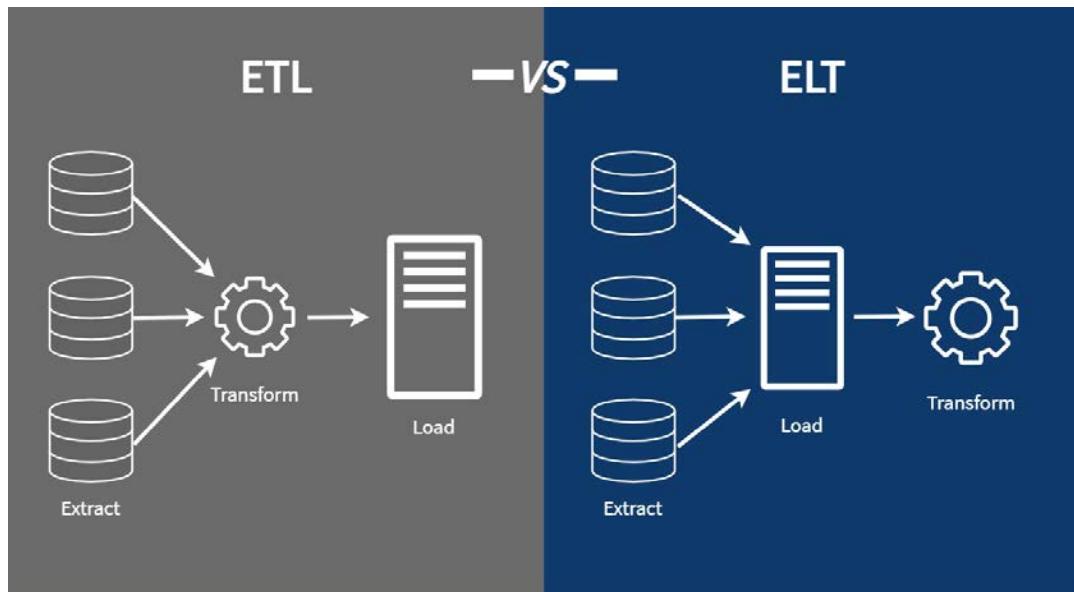
- Cloud computing power
- Latest innovation
- Security
- Compliance
- Faster performance

- Flexibility

## ▼ Cloud's Role in Advancing Data Analytics

### Data Integration

Cloud data can be managed through **data integration** or **data ingestion**. Data integration combines information from various sources into a unified, usable data source. This process can be achieved through two main methods:



- **ETL (Extract, Transform, Load)**
- **ELT (Extract, Load, Transform)**

Both ETL and ELT are cloud-based approaches that utilize the power of data warehouses, such as **Google BigQuery**, to transform data. The key difference between these two processes is:

- **ETL** transforms the data before loading it into the warehouse.
- **ELT** transforms the data after it has been loaded into the warehouse.

Either method ensures that the data is ready for further analysis or processing.

### Data Ingestion

**Data ingestion** involves the collection, importation, and processing of data for later use or storage. This process can be done in two ways:

- **Stream Ingestion:** Real-time continuous processing of data as soon as it is collected from various sources.
- **Batch Ingestion:** Data is processed at predefined intervals or in larger chunks.

Both methods have transformed how organizations can interact with and access data efficiently.

## Tools and Technologies Supporting Cloud Data Analytics

There are many tools that have enhanced cloud data analytics and how users access and manage their data. These include:

- **Web Interfaces**
- **APIs (Application Programming Interfaces)**
- **SQL** for querying data
- **Ingestion tools** such as **Pub/Sub (Kafka Alternative)**
- **Business Intelligence Solutions** like **Looker** and **Jupyter Notebooks**

These tools allow users to access their data stored in the cloud anytime and from anywhere, enabling seamless workflows and enhanced collaboration.

## The Impact of Cloud Data Analytics on Business Processes

The evolution of cloud data analytics has greatly benefited various **data analysis activities**. Some key areas include:

- **Big-data analysis**
- **Asynchronous data visualization** from multiple sources
- **Artificial Intelligence (AI) and Machine Learning (ML)**
- **Custom report generation and analysis**
- **Data mining and data science**

Cloud analytics has made these processes more accessible, faster, and cost-effective, allowing organizations to leverage powerful tools and insights.

## ▼ Cost Considerations of Cloud Services

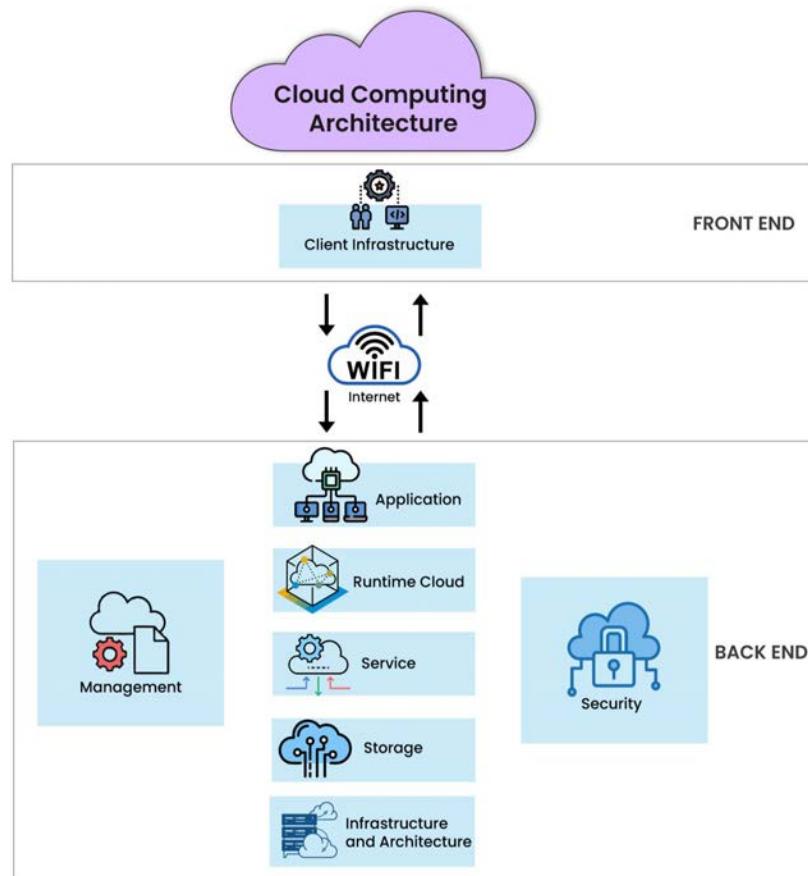
### Cloud costs:

- Resource provisioning
  - This is the process of a user selecting appropriate software and hardware resources and the cloud service provider setting them up and managing them while in use.
  - The resource provisioning process occurs through one of three delivery models:

- advanced provisioning
    - In advance provisioning, the user signs a formal agreement with the cloud service provider and either pays a set price or is billed monthly.
    - Then, the provider gathers the agreed upon resources and delivers them to the user.
  - dynamic provisioning
    - In dynamic provisioning, resources are adjusted based on the user's changing needs and they're only charged for what they use.
    - This means they can easily scale up or down based on usage demands.
  - self-provisioning.
    - With self-provisioning, also known as cloud self-service, the user purchases resources from the cloud provider through a website or online portal, and then the resources are quickly made available for the user, usually within hours or even minutes!
- Storage
    - Storage is ranked as one of the top three cloud expenses for many organizations, and the demand for more storage capacity only continues to grow.
    - Storage costs vary based on data storage, data processing, and network use.
  - Running queries

## ▼ Overview of Cloud Architecture

### ▼ Cloud Architecture Components

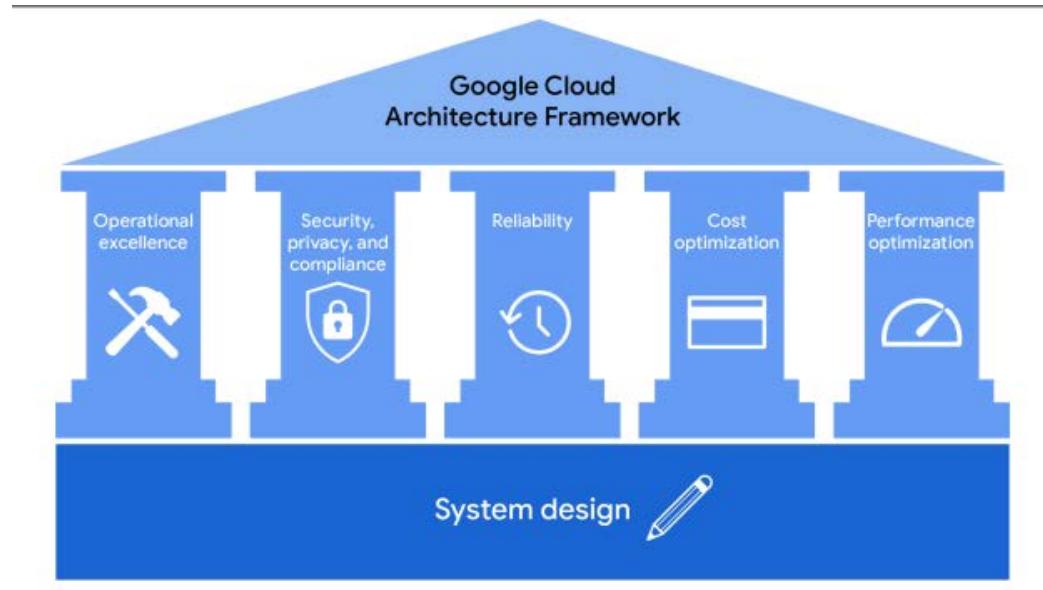


- **Frontend platform**
  - Frontend platforms include the parts of the architecture that users interact with: the screen design, apps, and home or work internet networks.
  - Frontend platforms allow users to connect and use cloud computing services
- **Backend platform**
  - Backend platforms are the components that make up the cloud itself, including computing resources, storage, security mechanisms, and management.
    - Backend component
      - **Application:** This behind-the-scenes software is accessed by a user from the frontend and then the backend component completes the task.

- **Service:** You can think of service as the heart of a cloud architecture because it takes care of all the tasks happening in the computing system and manages the resources users access.
  - **Runtime cloud:** Runtime cloud provides a space for all cloud services to perform efficiently. It's similar to your laptop or phone's operating system and ensures cloud services run smoothly.
  - **Storage:** This is where cloud service providers keep the data to run systems. Different cloud service providers offer flexible and scalable storage designed to hold and organize vast amounts of data in the cloud.
  - **Infrastructure:** It's made up of important hardware that allows the cloud to work, and it keeps systems at their best, including the central processing unit or CPU, the graphics processing unit or GPU, and the network devices.
  - **Security:** Planning and designing security for data and networks provides critical supervision of systems, stops data loss, and avoids downtime.
- **Cloud-based delivery model**
  - **Network**

## ▼ Cloud Architecture Framework

The six pillars of the Google Cloud Architecture Framework include: system design; operational excellence; security, privacy, and compliance; reliability; cost optimization; and performance optimization. Each of these pillars represent design considerations that can guide your work in the cloud.



1. System design is the foundation of the Google Cloud Architecture Framework. When designing your system, you must define the architecture, components, modules, interfaces, and data needed to satisfy cloud system requirements, as well as continue to learn about the products and features that support system design.
2. Operational excellence determines how efficiently you deploy, operate, monitor, and manage your cloud workloads.
3. Security, privacy, and compliance involves ensuring your data is secure in the cloud, your information is private, and that your design aligns with organizational standards.
4. Reliability means that your system is designed to handle your workloads in the cloud.
5. Cost optimization maximizes your business investment in cloud architecture for your organization.
6. Performance optimization means that, in your design, you are continuously honing your cloud resources to promote optimal performance.

## ▼ The Data Lifecycle

### ▼ Introduction to Data Management

## What is Data Management?

**Data management** refers to the process of establishing and communicating a clear plan for collecting, storing, and using data. A **data management plan** ensures that

all employees follow consistent procedures across these steps. In some organizations, this may also be referred to as **data governance**.

## The Importance of a Data Management Plan

There are three main reasons why data management is essential for organizations:

### 1. Seamless Collaboration:

A proper plan ensures that all users can access data using documented procedures, maintaining governance and compliance with internal and external regulations.

### 2. Data Security:

It sets clear parameters to protect data from breaches, unauthorized access, or accidental data loss.

### 3. Scalability:

A well-structured plan makes it easier to scale operations, with defined procedures for handling growing amounts of data.

As a **cloud data analytics professional**, you may either create such plans or work within one that's already in place. Key aspects to consider include **access, data types, storage, and archives**.

## Key Aspects of a Data Management Plan

### 1. Access

A data management plan defines who can access the data and the level of access each user has. This can be as detailed as specifying access rights to certain rows or columns in a dataset. For example, only the marketing department may have access to users' email addresses to send notifications, while only analysts can access users' birthdates to generate age-segmented reports.

### 2. Data Types

The plan should clearly outline the types of data the organization is allowed to collect, such as **Personally Identifiable Information (PII)**. It should also specify which teams have access to this data and for what purposes.

### 3. Storage

A good plan should include the storage systems used, such as **Google BigQuery** projects or **Google Drive** folders. It should also account for backup strategies in case of outages.

### 4. Archives

Finally, the data management plan should establish procedures for archiving or deleting data, in line with both internal business guidelines and external regulations. For instance, data used in legal proceedings may be exempt from standard archiving rules.

## ▼ Safety and Privacy in the Cloud

### What is data privacy?

Data privacy is preserving a data subject's information any time a data transaction occurs.

### Personally identifiable information (PII)

PII is data that is unique to an individual and therefore can be used to distinguish one person from another.

- A user's email address, mailing address, phone number, precise location, birthday, and full name are all examples of PII and must be safeguarded.

### Protected health information (PHI)

Along those same lines, personal health data is PHI, or protected health information.

- PHI is health data that can identify an individual, like information about patient demographics, mental or physical health diagnoses or treatments, and payment records related to health care.

## ▼ Stages of the Data Lifecycle

This is the sequence of stages that data experiences, which include: plan, capture, manage, analyze, archive, and destroy.

Data lifecycle phases mark the journey that data takes from start to finish of any data project. Here's a list of all the phases and what they mean:

- Plan: The planning stage begins before you start any analysis, and involves answering a business question or meeting an objective.
  - Business question
  - Objective
  - Data types
  - Data management process
  - Who is responsible for each stage
  - Outcomes

- How to measure success

**Pro tip:** The planning stage should include deciding what type of data to collect, how it will be managed, and who is responsible for each stage of the lifecycle.

- Capture: In the capture phase, data is collected from different sources, and gaps are identified.
  - Data is collected
  - Gaps are identified
  - Iteration occurs
  - Process captured data: Clean, transform, compress, encrypt
- Manage: The manage phase ensures that data is properly maintained, and stored safely in a secure location.
  - data maintenance.
  - safe and secure data storage
  - ongoing process
- Analyze: In the analyze phase, analysts use data to answer the business question or help meet the business objectives.
  - answer the business question
  - meet the business objective
- Archive: In the archive phase, data is stored for later use, and for compliance purposes.
  - store data for later use
- Destroy: In the destroy phase, the data that is no longer useful to the organization is permanently deleted.
  - when data is no longer useful



## **Data lifecycle vs Data analysis cycle**

- Data lifecycle: how the data itself moves and changes throughout its existence
- Data analysis process: how data analysts interact with the data

## **▼ Importance of Automation in the Data Lifecycle**

Data management:

- Ingest data on a set schedule
- Automatically cap the amount of data ingested
- Be more efficient with a set amount of resources

Automation is the use of software, scripting, or machine learning to perform data analysis processes without human work.

Benefits of automation:

- Reduces mistakes
- Checks for errors or incomplete data
- Ensures uniformity and accuracy
- Scale up workloads
- Allows for efficiency and process improvement

## ▼ Introduction to Data Retention Policies

| Data retention is the collection, storage, and management of data.

This is a key part of data management, which deals with how and when data is saved or deleted.

## ▼ Overview of Version Control and Holds

| Versioning is the process of creating a unique way of referring to data.

Benefits of versioning:

- Support quality control
- Revert to a previous version
- Support compliance concerns
- Find where edits are incorrect

| A hold is a policy placed on a dataset that prevents its deletion or prevents deletion capabilities for certain accounts.

Benefits of holds:

- Prevent accidental deletion
- Preserve data indefinitely
- Determine and define each user's responsibility for datasets

## ▼ The Role of a Cloud Data Analyst

### ▼ Data Analysis in the Cloud

Cloud data analytics is the process of analyzing and extracting meaningful insights from large volumes of data using cloud-based services and solutions.

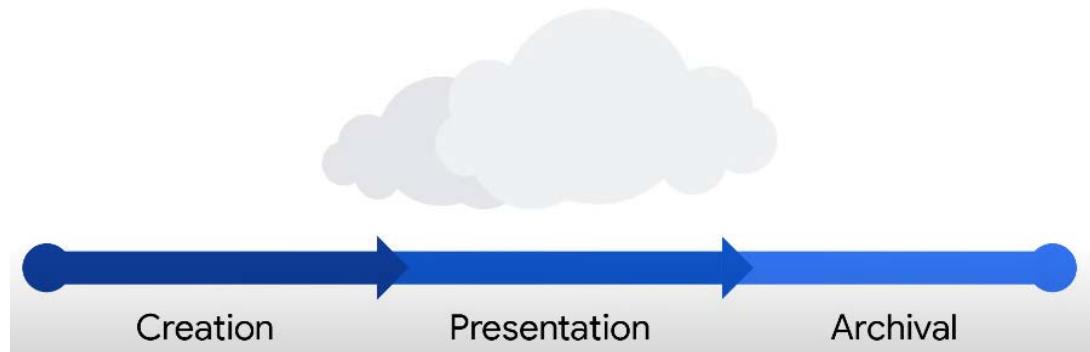
Cloud data analytics allows analysts to access large amounts of data from multiple sources without having to worry about setting up the infrastructure and security for each dataset.

### Advantages of working with data in cloud

- Quick and easy access to real-time data from different sources.

- Analyzing large datasets is simplified because computing and processing power don't rely on local machines.
- Data professionals can aggregate and analyze data right in the cloud.

With cloud databases and tools, you can automatically ingest large amounts of data and train your system to generate themes and summaries. Much of this is made possible by a cloud-based data pipeline.



Through that pipeline, information moves seamlessly from creating data to archiving data.

## ▼ Overview of Google's Cloud Data Analytics Tools

### **BigQuery**

- A serverless data warehouse
- This tool is pretty popular in the data world because it works across many major cloud platforms.
- You use SQL to query and organize data
- ML and AI tools
- BigQuery includes built-in business intelligence engines to create interactive and responsive data insights.

### **Looker**

- Looker organizes business data and builds workflows and applications based on data insights.
- Primarily a data visualization and reporting tool.
- You might use Looker to integrate various file types like CSV, JSON, and Excel into a single application.
- You may also use it to publish data in a variety of dashboards.

## **Dataproc**

- This service is fully managed and allows you to run Apache Hadoop, Apache Spark, and Apache Flink, along with many other open-source tools and frameworks.
- Dataproc can modernize data lakes and perform ETL functions —or extract, transform, and load— for massive amounts of data.
- You might use Dataproc to process large data workloads and place them in clusters that allow you to access your data quickly.

## **Dataflow**

- Which gives you the ability to stream and batch process data in a serverless application.
- Dataflow can be used to develop data processing pipelines.
- It can assist with reading the data from its original source, transforming it, and writing it back to your database.

## **Cloud Data Fusion**

- Another fully managed service that allows you to integrate multiple datasets of any size.
- You might use Cloud Data Fusion because it allows you to use a graphical user interface instead of code to manage your data pipelines.

## **Dataplex**

- You to work with multiple data sources.
- It creates a central hub for managing and monitoring data to work across various data lakes, data warehouses, and data marts.

## **BigLake**

- BigLake is a storage engine that you can use to unify data warehouses and lakes with BigQuery and open-source frameworks.
- It also provides options for access control, and multi-cloud storage.

## **▼ Data Access Management for Secure Data**

Data access management is the process of implementing features including password protection, user permissions, and encryption to protect data and related processes. In the context of the cloud, it includes the creation of individual user roles or user groups. And the access for each role or group is pre-defined by an

administrator. The act of assigning this access is called identity access management, or IAM.

## **Identity Access Management (IAM)**

IAM is the process of assigning certain individual roles or groups with access to particular resources. And some key resources may include software hosted in the cloud, datasets, databases, or entire data warehouses. The level of access is determined by the role of the user. There are three main components of identity access management:

- Principal: This is the account or accounts that can access a resource.
- Role: This is the collection of permissions that lists the elements that an account has access to.
- Policy: This is the collection of roles that are attached to a principal.

## **Best practices in data management**

- Audit data access and monitor user tasks.
- Put additional permissions on remote access.
- Verify and stop unusual activities.
- Add two-factor authentication or strong passwords to accounts to ensure security.

## **▼ Introduction to Business Data Requests**

A business data request is any business question that can be answered with data. Whether the request involves data that already exists, or new data. As a data analyst, you'll typically receive this request through a ticketing system. This is a tool for recording requests and assigning them to the appropriate team member. Data requests can be simple or complex.

### **Elements of data requests:**

- Can be internal, from employees within the organization, or external, from outside.
- Types of requests might include answers to a data-related question or the creation of a data report, an extract, or a dashboard.
- Requests usually include information or structure specifics.

### **Working on a data requests:**

- Overarching goal

- What's being measured
- What data is needed and how much
- Outliers
- Trends
- Summary reports

### ▼ Accurate Data Ensuring Tips:

Data analysis requires curiosity, attention to detail, and persistence. However, sometimes the data you have may not be as useful as expected. Here are strategies to ensure accurate data:

- **Clarify the Request:** Start by thoroughly understanding the business question, its goals, and the expected outcomes.
- **Assess Available Data:** Use analytical tools to check the database structure. Ensure you have the necessary data points, such as timestamps for key user actions (e.g., searching for products or making purchases).
- **Data Cleaning:** Eliminate inaccurate or duplicate records. For example, if a customer submits duplicate orders to correct a mistake, remove the repeated data. Use this phase to segment and organize data meaningfully.
- **Identify Outliers:** Check for any outliers (e.g., mistakenly ordering 1,000 items instead of 10). Address these deviations to maintain data integrity.
- **Integrate Multiple Sources:** Use data from various systems (e.g., website, CRM, purchase orders) to cross-verify accuracy.
- **Leverage Summary Statistics:** Run statistical summaries to check for consistency across averages, medians, and other key metrics. Use SQL queries or dashboards to do so.
- **Compare Past Analyses:** If similar analyses have been done, compare them. Consistent results indicate reliable data, while large discrepancies suggest the need for revalidation.

### ▼ Cloud Data Storage and Management Tools Summary:

Cloud-based data analytics provides a range of tools to manage and store data efficiently. In this scenario, you're a data analyst working for a gaming company, tasked with analyzing how many users click on ads during gameplay. Here's how various cloud tools assist in this process:

- **Google Cloud Storage:** Used to compile and store data from multiple sources. It allows you to upload data from remote servers, centralizing your databases and analytics tools.

- **Dataflow:** Helps create a data pipeline for both stream and batch processing. You can stream live data from your apps into a single database or import existing data in batches.
- **Cloud Data Fusion:** Enables you to build and manage high-volume data pipelines, continuously integrating user data from your applications.
- **BigQuery:** Used to run SQL queries to clean and join data. This ensures that your data is complete, free of duplicates, and ready for analysis.
- **DataProc:** Allows you to leverage open-source analytics tools and apply large-scale programming languages and algorithms for data analysis.

By using these tools, you can seamlessly manage, store, and analyze data, providing valuable insights to stakeholders. In this case, the advertising team will use your findings to decide whether to continue displaying ads or switch to a subscription model for revenue.

## ▼ What is Dataproc?

Dataproc is a fully managed service by Google Cloud that supports open-source tools like Apache Hadoop for big data processing. It's ideal for handling both structured and unstructured data. Here's a breakdown of Dataproc's key features and how it's used:

### 1. Scalability and Efficiency

- **Dynamic Resource Management:** Dataproc allows you to scale resources up or down as needed, so you don't have to guess how much computing power is required.
- **Cost Savings:** If data processing isn't active, only the storage services remain running, preventing unnecessary compute costs.

### 2. Integration with Hadoop

- **Fast Setup:** Dataproc can set up a Hadoop cluster in just **90 seconds**, enabling you to process large-scale data efficiently.
- **Real-Time Processing:** You can leverage multiple computers in parallel to process data in real-time, providing faster insights.

### 3. Use Cases

- **Multiple Data Sources:** Dataproc can process data from various sources, such as customer reviews or sales information, helping businesses make informed decisions.

- **Comprehensive Analysis:** Once processed, data can be sent to Google Cloud Storage (GCS), BigQuery, or data science notebooks for further analysis, offering flexibility for different data needs.

## 4. Pre-Built Templates for Easy Migration

- **Data Migration:** Dataproc offers pre-built templates that make it easy to move data from your existing infrastructure to Google Cloud. For instance, you can automatically transfer data from platforms like **Snowflake**, **Redshift**, or **S3**.

## 5. Security and User Permissions

- **Advanced Security:** Dataproc integrates with organization-wide security systems, allowing for better data control.
- **User Permissions:** You can assign different levels of access to various users, ensuring that data management is both secure and well-organized.

Dataproc streamlines data processing, reduces costs, and enables businesses to analyze large datasets effectively. For example, an online retailer can use Dataproc to provide real-time insights to suppliers, helping them optimize product offerings based on customer demand. This makes Dataproc a powerful tool for businesses looking to stay competitive in the fast-paced digital world.

## ▼ Introduction to Process Management

Data analysts often handle multiple projects simultaneously, each in different stages, with varying tasks and team involvement. Managing this requires juggling skills—keeping several tasks progressing without dropping any.

To master this, you need an organized internal process workflow. There are three key components to a typical data analyst workflow:

1. **Data Request Central System**
2. **Checking in Code**
3. **Keeping an Internal Record of Work**

### 1. Data Request Central System

This system stores and manages business data requests and team discussions, allowing easy access to:

- **Documentation:** Preserve details and conversations about each request.
- **Collaboration:** Teams review data requests and work on them together.
- **Historical Records:** Access past information when needed.

## 2. Checking in Code

For teams writing code, it's important to use a central repository (e.g., GitHub).  
Code check-ins:

- Allow teammates to access, review, and approve code changes.
- Improve collaboration, reduce errors, and ensure consistency in reports or dashboards.

Check-in benefits:

- More effective collaboration
- Centralized repository
- Improve code quality
- Track code easily
- Easier way to revert code
- Revision history
- Easier to locate code

## 3. Keeping an Internal Record of Work

Logging tasks in a central place organizes your workflow, ensures team progress visibility, and helps balance multiple projects.

By managing requests, code, and tasks effectively, you can maintain a smooth process and deliver quality results.

### ▼ Strategies for Handling Data Requests

#### What is a Business Data Request?

A business data request is any question or inquiry that can be answered with data. Stakeholders often submit these requests through a **ticketing system**, which tracks and organizes them. A data analyst uses the same system to prioritize and manage the requests.

#### Key Elements of Data Requests:

- **Type:** Grouping similar requests.
- **Priority:** Determining the urgency of each request.
- **Status:** Updating the current progress.

#### Data Requests:

- Reports
- Data clarifications
- Reference data
- Data extracts

## Gathering Details:

When receiving a request, gathering comprehensive details upfront can help reduce back-and-forth later. To ensure clarity, ask the following questions:

1. **What:** Define the scope of the data. What exactly should be included?
2. **When:** Determine the time frame of the data—Is it for this year or a historical period?
3. **Who:** Clarify the subjects or groups involved—Are all data points or just a subset required?
4. **Where:** Understand the geographic or other segmentation of the data.
5. **Why:** Grasp the business context of the request—What questions are the stakeholders trying to answer?
6. **How (Data Refresh):** Decide whether the data needs to be updated regularly or is it a one-time pull?
7. **How (Data Delivery):** Determine the format—Will it be a simple report or a dynamic dashboard?

## Handling Complex Requests:

Some requests require breaking down into smaller tasks, especially if they involve dependencies. **Ticketing systems** allow for creating **parent-child relationships**, where subtickets are generated for specific parts of the larger request. This enables multiple team members to work on different parts simultaneously.

## Tracking Status:

Each ticket in the system has status fields like **assigned**, **in progress**, **fixed**, and **verify**, providing transparency on the current stage of each request. This helps teams stay organized and ensures stakeholders are updated on the progress.

## Benefits of an Efficient System:

By efficiently managing data requests through a centralized system, you can:

- Improve communication and collaboration.
- Track requests from submission to completion.

- Provide quick and accurate responses to data needs.

## ▼ Clear Documentation Benefits the Whole Team

Welcome! In this video, we'll explore the vital role of **data documentation** and how it keeps data team members aligned and efficient.

### What is Data Documentation?

Data documentation serves as a written guide to the datasets, detailing how the data was collected, organized, and validated. It typically includes:

- The purpose of data collection.
- Procedures followed during data gathering.
- Date and time of data collection.
- Structure of the dataset.
- Notes on data validation or quality assurance.

Documentation can take various forms, such as:

- **Readme files**
- **Data dictionaries**
- **Codebooks**
- **Lab notebooks**
- **Spreadsheets**

### The Importance of Data Documentation

Let's illustrate this with an example. Imagine you've recently joined a data team at an innovative renewable energy company. As you start working, you notice inconsistencies:

- Wind and geothermal data are recorded in separate spreadsheets with different structures.
- Different departments maintain their own data tables, leading to integration challenges.
- Your team only has access to the last two years of data, despite ten years being available.

Fortunately, a coworker hands you the team's **data playbook**. This document outlines the data management plan and provides clarity on:

- How to request and grant data access.
- Where to find data tables (e.g., in BigQuery).

- Common tasks and sample queries for data operations.

## Real-World Application

As you delve into the playbook, you learn:

- Why the wind and geothermal data are kept separate—there are specific projects crucial for environmental impact.
- Examples of successful data merging from past initiatives, which help you see how to approach your current tasks.

During a team retrospective, you discuss the playbook's benefits and learn that it is a **living document**, regularly updated to reflect changes and improvements. Your team values contributions to this documentation, ensuring that everyone is informed and aligned.

## Conclusion

In conclusion, effective data documentation, like a well-maintained data playbook, is essential for:

- Keeping everyone on the same page.
- Streamlining data requests and access.
- Enhancing collaboration and efficiency within the team.

Having up-to-date documentation ensures that you and your data team can navigate the complexities of data management successfully!

# ▼ 2- Introduction to Data Management and Storage in the Cloud

## ▼ Introduction to Data Management and Storage in the Cloud

### ▼ Introduction

Welcome to this exciting course on **cloud storage and data management in the cloud!**

#### What You'll Learn:

- **Data Storage:** Understanding data connections, types, and structures.
- **Table Schemas:** Handling batch and streaming data processing.
- **Denormalized Data:** Learning about data governance, metadata, and data catalogs.

- **Data Lakehouse Architecture:** How to store, process, and analyze large datasets effectively.
- **Advanced Tools:** Explore **Dataplex** for identifying data sources in **BigQuery** and learning to manage tables, add/export data, and query in **BigQuery**.
- **Google Cloud Services:** Access data from various services and manage a **Dataproc** cluster.
- **Data Partitioning:** The benefits and implementation for efficient data organization.

This course is packed with hands-on lessons designed to give you a **strong foundation in data organization**, which is an essential skill every employer values. You've made great progress in your data analytics journey so far, but there's much more to explore. Let's keep the momentum going!

Let's dive in and start learning!

## ▼ Data Storage and Connections

Imagine this scenario: You're casually browsing a website, and suddenly, you see an ad for a product you've never considered before but now, it's exactly what you need! It could be something like a **tea kettle disguised as a houseplant**—how did they know you'd love it?

Most likely, a business has tracked your previous purchases and used that data to send personalized ads, offers, and coupons. As a data analyst, managing this level of personalized information for **millions of customers** is a significant task.

Businesses need to collect and analyze data from various sources and formats to recommend the right products for their customers.

This is where **data storage** and **data connections** come in, and that's what we'll explore in this topic!

## Cloud Data Storage

Cloud data storage allows organizations to store, access, and manage digital data in **off-site servers** hosted by cloud providers. Businesses don't need to own or manage these servers, and the data stored can include anything from files and business data to videos and images.

As a data analyst, your focus will primarily be on **input and output data**:

- **Input data:** Information a user sends to a computer (e.g., typing text, filling out an online form, or scanning an image).
- **Output data:** Information a computer sends back to the user (e.g., generating a pie chart from a spreadsheet or producing an order summary).

## Data Storage Systems

There are several common storage systems used to manage input and output data:

1. **Systems of Record:** These serve as the **source of truth** for an organization's data related to processes or systems. All data is uniform, and the system of record is the authoritative source for critical data.
2. **Transactional Databases:** These databases store each **transaction** as an individual row. Commonly used in **e-commerce** and **online banking**, each transaction (e.g., a customer purchase) creates a new row in the database containing details such as customer name, purchase date, and sales price. A **transactional database** can also be a system of record.
3. **Cloud Storage:** This solution allows organizations to store digital data on **cloud-based servers** hosted by a cloud provider. It's scalable, meaning the capacity can increase or decrease as needed.

## Data Connections

A **data connection** is a link between a data source and a tool that allows you to access and interact with that data. For example, if you want to create dashboards or reports, you'll need tools that can connect to your **transactional databases**, **cloud storage**, or other storage systems.

By using **data connectors**, business intelligence or analytics tools can access multiple data sources, enabling you to create analyses, pivot tables, or other reports from a single platform.

## ▼ Common Ways to Store Data

The document introduces the importance of data organization, emphasizing its critical role in the data profession and how it helps optimize time management. With the sheer volume of data generated globally—about 2.5 quintillion bytes or 2.5 billion gigabytes daily—the need for effective data storage solutions is more essential than ever. The document then covers various common methods of storing data, each suited for different use cases.

1. **Relational Database:** A widely used system that organizes data into related tables, allowing complex queries across multiple datasets. This structure is essential for businesses as it enables efficient data management and querying.
2. **Data Warehouse:** This is a centralized database designed for consolidating data from multiple sources. It ensures data consistency and is optimized for complex analytical queries, making it a go-to solution for organizations requiring detailed data analysis.

3. **Data Mart:** A more focused subset of a data warehouse, data marts are subject-oriented and are used by specific departments or areas within a business. They provide streamlined access to relevant data, making them ideal for business intelligence and reporting tasks.
4. **CSV File:** A simple format using commas to separate values, typically used for smaller datasets or spreadsheet-like data. However, CSV files are not scalable and are inefficient for large or complex datasets.
5. **Data Lake:** A system that stores vast amounts of raw data in its original format. Data lakes are highly versatile, capable of handling diverse sources and data types without limitations in size or structure, making them ideal for big data environments.

In conclusion, understanding the different types of data storage tools is essential for effective data management. Whether it's a relational database for detailed queries or a data lake for large-scale data storage, each method offers unique benefits depending on the nature of the data and the analytical requirements.

## ▼ Structured, Unstructured and Semi - Structured Data

There are three main types of data storage: structured, semi-structured and unstructured data. Understanding these types is essential for data professionals to ensure efficient data processing and querying for different projects.

1. **Structured Data:** This type of data is highly organized, typically arranged in rows and columns, such as in spreadsheets or SQL databases. Structured data is easy for computers to process because it adheres to a defined schema. SQL databases, for example, use tables with primary keys (unique identifiers) and foreign keys (used to link related tables) to create relationships between datasets. This structured approach allows for complex queries across multiple tables and is ideal for storing data that follows a predictable, repeatable format.
2. **Unstructured Data:** In contrast, unstructured data lacks any predefined organization, making it more difficult to process. This data type can include text, images, videos, and other forms that don't fit neatly into tables or predefined formats. Social media posts and multimedia content are common examples of unstructured data. Tools like cloud-based data lakes (e.g., Google Cloud Storage or Google Cloud DataProc) are designed to handle unstructured data, offering flexibility for storing, retrieving, and analyzing it without requiring a fixed schema. This type of data can be valuable for a wide range of purposes but poses challenges in interpretation and storage.
3. **Semi-structured Data:** This data type falls between structured and unstructured, containing elements of both. Semi-structured data is more flexible than structured data but still has some organizational framework, making it easier to work with than unstructured data. A typical example is an

email: while its header contains structured data like the sender, recipient, and date, the body of the email (which can include text, images, and attachments) is unstructured. Semi-structured data is commonly used in situations where data doesn't fit neatly into traditional databases but still requires some level of organization.

In summary, recognizing these three types of data storage methods—structured, unstructured, and semi-structured—is crucial in determining how to effectively manage, store, and query data. Each type offers distinct advantages depending on the project's requirements, and mastering them will enhance any data-driven decision-making process.

## ▼ Example of a Data Lakehouse

This document provides a comprehensive overview of the evolution from traditional data warehouses to data lakehouses, emphasizing the benefits and challenges of each system.

1. **Data Warehouse Limitations:** Initially, businesses relied on data warehouses, which were centralized and structured storage systems, to manage and analyze data. However, as businesses began collecting large volumes of unstructured and semi-structured data (e.g., text, images, videos, and sensor data), traditional warehouses struggled to efficiently store and process these diverse datasets.
2. **Emergence of Data Lakes:** To address these limitations, data lakes were introduced. Unlike warehouses, data lakes can store vast amounts of raw, unstructured, and semi-structured data. While this allowed for more flexible data storage, it also presented challenges in data governance, quality control, and performance.
3. **Introduction of the Data Lakehouse:** The data lakehouse architecture was developed as a hybrid solution, combining the strengths of both data warehouses and data lakes. A lakehouse offers a unified platform for storing, processing, and analyzing both structured and unstructured data. It also supports scalability, seamless integration with analytics tools, and improved data management.
4. **Business Example:** Consider an agricultural company that initially used separate infrastructures for its on-premise data warehouse and cloud-based data lake. This separation led to inefficiencies in data processing and difficulty extracting timely insights. The company's data warehouse struggled to handle large volumes of diverse data, such as JSON documents, call logs, and survey transcripts, often resulting in system crashes. Meanwhile, the data lake lacked structure, making data quality and discovery problematic.

5. **Migration to a Data Lakehouse:** To overcome these challenges, the company migrated to a data lakehouse. This transition allowed them to store and process raw data efficiently while introducing the necessary structure for effective analysis. The lakehouse also enabled scalable resource management during peak periods, ensuring optimal performance.
6. **Governance and Advanced Analytics:** The data lakehouse also improved the company's data governance by implementing defined schemas, access controls, and security measures. This enhanced governance boosted data quality, ensured compliance, and instilled confidence in stakeholders. Additionally, the company integrated advanced analytics and machine learning tools, allowing data scientists to explore large datasets, experiment with algorithms, and generate actionable insights.
7. **Results:** The move to a data lakehouse significantly improved the company's agility in reporting and responding to market changes. With better data governance and security, the company ensured the privacy of sensitive information while enhancing decision-making processes.

## ▼ Aspects of table schema

1. **Definition and Importance of Schemas:** Schemas are essential organizational tools that help individuals make sense of various systems, similar to how a museum schema provides an overview of what to expect. In data management, schemas are critical for understanding how tables are structured.
2. **Components of a Table Schema:**
  - **Column Names:** Each column in a BigQuery table has a unique identifier known as a column name, which can consist of letters, numbers, and underscores. They must begin with a letter or underscore, and uniqueness among column names is crucial for ease of access during queries.
  - **Data Types:** The data type of a column defines the kind of data it can store. BigQuery supports various data types, including:
    - **Number Types:** For storing numeric data, such as integers and floating-point numbers (e.g., numeric, integer, float64).
    - **String Types:** For storing text data and binary data as bytes.
    - **Date and Time Types:** For storing date and time information, like order dates.
    - **Location Types:** For geographic coordinates (e.g., latitude and longitude).
    - **Time Interval Types:** For periods of time, such as fulfillment durations.

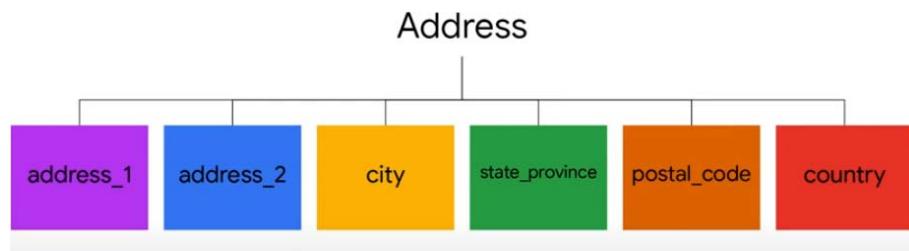
- **Complex Data Types:** BigQuery allows for arrays (repeated columns) to store multiple values of the same type, and structs (records) to store values of different types within a single column. JSON objects can also be utilized for key-value data storage.

3. **Mode:** The mode indicates whether a column can contain empty values. A nullable column allows for empty values, while a required column does not. Additionally, setting the mode to "repeated" allows a column to function as an array.
4. **Usage in Querying and Data Management:** Understanding the structure of a table through its column names, data types, and modes enables users to write effective queries, group data, create tables, and share data efficiently.

In conclusion, mastering the aspects of table schemas in BigQuery is vital for optimizing data organization and streamlining workflows, positioning users to become proficient in managing and querying data.

## ▼ Introduction to nested structure

1. **Introduction to Nested Structures:** Nested data structures are used to organize data within other structures, creating a hierarchy similar to a company's organizational chart or a website's page structure. They play a crucial role in representing relationships such as parent-child associations and multi-level categorizations, facilitating advanced data analysis.
2. **Types of Nested Data:**
  - **Structs (Records):** A struct groups related columns together, making data easier to organize and query. For example, a customer's address, including fields like city, state, and postal code, can be grouped into a single struct called "address."



- **Arrays (Repeated Columns):** An array is a list of values of the same data type stored in a single column. Arrays can contain numbers, strings, structs, or even other arrays. They are useful for storing lists, such as a customer's multiple email addresses or an array of structs for different types of addresses (e.g., home, work, billing).



### 3. Benefits of Nested Data:

- **Efficient Data Management:** Nested structures help maintain data integrity by organizing related data within the same structure, making it easier to process, store, and retrieve information.
- **Advanced Analysis:** Nesting allows analysts to capture complex relationships in data, supporting more sophisticated queries and analyses.

### 4. Querying Nested Data:

Querying nested data differs from querying flat structures. To work effectively with nested data, one must understand how the nested elements relate to each other and how to structure queries that interact with them.

In summary, nested data structures, such as structs and arrays, provide flexible and powerful tools to manage complex datasets, enabling more efficient data analysis and improved organization of related information.

## ▼ Overview of Data Processing Methods

1. **Batch Processing:** Batch processing involves collecting large volumes of data over a period of time and then processing it all at once. It's typically used for structured formats like CSV, JSON, and Parquet files. The most common formats include CSV, or comma-separated values, found in spreadsheets and many types of databases; JSON, or JavaScript Object Notation, which provides easy-to-read representation of complex data structures; and Parquet, a columnar storage format for quick compression and improved query performance. This method is efficient for large-scale data warehousing, complex analytics, and processing historical data for long-term planning.

However, generating insights can take time as data is processed in intervals, ranging from hourly to weekly, depending on the use case.

**2. Streaming Data Processing:** Streaming data is processed in real-time or near-real-time, continuously as it is received. This method is essential for time-sensitive applications, such as monitoring live IoT devices, social media feeds, or real-time event detection. File formats like Avro and tools such as Apache Kafka and Apache NiFi are designed to handle the high velocity and agility of streaming data. Streaming allows organizations to monitor live data streams, detect anomalies, and trigger automated responses based on predefined conditions, making it invaluable for scenarios requiring immediate action.

- **Avro:** Avro, which is a file format that helps programs understand and share data using schemas and seamless handling of data structure changes.
- **Apache Kafka:** Apache Kafka, an open-source distributed event streaming platform that allows you to publish, subscribe, store, and process streams of records. This is particularly useful for building data pipelines and applications that require handling large amounts of data.
- **Apache NiFi:** Apache NiFi, an open-source data integration and data flow automation tool which is particularly useful for collecting, transforming, and moving large volumes of data.

**3. Advantages and Disadvantages:**

- **Batch Processing** is ideal for processing large datasets with complex operations (e.g., aggregations, transformations), especially in analytics or historical data processing. However, it may be slower and not suitable for real-time insights.
- **Streaming Data** enables near-real-time insights, continuous monitoring, and timely responses to critical events. While powerful for real-time applications, it tends to be more complex and resource-intensive to implement and maintain.

**4. Decision Factors:** When choosing between batch and streaming processing, consider factors like data characteristics, processing speed, latency requirements, and integration with existing systems. Both paradigms offer significant advantages, and selecting the appropriate method depends on the specific needs of the organization.

In summary, understanding the strengths and limitations of batch and streaming data processing is crucial for cloud data professionals to optimize data processing strategies and improve organizational efficiency.

▼ **Identify different batch and streaming data sources**

# Activity overview

Timely access to data is critical for organizations to respond quickly to market changes, customer needs, and operational issues.

Batch processing is a method of collecting large volumes of data over a period of time, then processing it all at once. It is best for processing large amounts of data and tasks that do not require near real-time processing.

Streaming processing is a method of processing data as it is received. It is best for processing data continuously in real time. It is important for data analysts and data scientists to understand the difference between the two since both processes have their own advantages and disadvantages.

As a data analyst, knowing when and how to apply batch processing or streaming processing helps you optimize the performance of data processing tasks, minimize time delay in data processing, and provide more accurate and timely data insights.

In this lab, you'll process and collect data for a specific purpose, and observe the data loading in BigQuery tables using both computer-assisted batch processing and stream processing methods.

## Scenario

You've been asked to help Meredith, the lead merchandiser at TheLook eCommerce, monitor the results of their promotions and price changes.

Merchandisers set prices and make sure they sell the inventory they buy. For example, if t-shirts for a sports team are overstocked in a championship year, a retailer might decide to lower the price so the merchandise sells faster. That way, the store does not end up with unsold stock.

For this task, Meredith needs to view the number of items added to shopping carts two ways. Near real-time monitoring will allow Meredith to view the number of items added to shopping carts as they are available.

Minute by minute monitoring will allow Meredith to view the number of items added to each shopping cart in increments of one minute.

This data will help Meredith track the effectiveness of their promotions and price changes over time. They can then use this information to help improve the shopping experience for their customers and increase sales.

Artem, the data architect, points out that shopping cart activity is streamed into one of the tables in BigQuery. You need to help Meredith understand how to find the data she needs to monitor their merchandise.

Here's how you'll do this task: **First**, you'll search for your dataset and associated table. **Next**, you'll run a query to display the time each product was added to the shopping cart. **Then**, you'll rerun the query. **Finally**, you'll examine the properties of the **shopping\_cart** and **order** tables.

## Setup

### Before you click Start Lab

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This practical lab lets you do the activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).

**Note:** Use an Incognito or private browser window to run this lab. This prevents any conflicts between your personal account and the Student account, which may cause extra charges incurred to your personal account.

- Time to complete the lab---remember, once you start, you cannot pause a lab.

**Note:** If you already have your own personal Google Cloud account or project, do not use it for this lab to avoid extra charges to your account.

### How to start your lab and sign in to the Google Cloud console

1. Click the **Start Lab** button. On the left is the **Lab Details** panel with the following:
  - Time remaining
  - The **Open Google Cloud console** button
  - The temporary credentials that you must use for this lab
  - Other information, if needed, to step through this lab

**Note:** If you need to pay for the lab, a pop-up opens for you to select your payment method.

2. Click **Open Google Cloud console** (or right-click and select **Open Link in Incognito Window**) if you are running the Chrome browser. The **Sign in** page opens in a new browser tab.

**Tip:** You can arrange the tabs in separate, side-by-side windows to easily switch between them.

**Note:** If the **Choose an account** dialog displays, click **Use Another Account**.

3. If necessary, copy the **Google Cloud username** below and paste it into the **Sign in** dialog. Click **Next**.

student-02-8ea7c3d1d192@qwiklabs.net

Copied!

content\_copy

You can also find the **Google Cloud username** in the **Lab Details** panel.

1. Copy the **Google Cloud password** below and paste it into the **Welcome** dialog. Click **Next**.

pUrytgNMSvHy

Copied!

content\_copy

You can also find the **Google Cloud password** in the **Lab Details** panel.

**Important:** You must use the credentials the lab provides you. Do not use your Google Cloud account credentials.

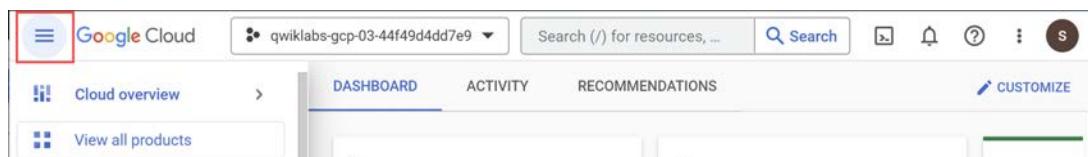
**Note:** Using your own Google Cloud account for this lab may incur extra charges.

1. Click through the subsequent pages:

- Accept the terms and conditions
- Do not add recovery options or two-factor authentication (because this is a temporary account)
- Do not sign up for free trials

After a few moments, the Console opens in this tab.

**Note:** You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.



# Task 1. Determine batch versus streaming sources

In this task, you'll help Meredith create a table that stores data about the number of orders and the time each order was added to a customer's cart.

1. In the Google Cloud console **Navigation menu** (), select **BigQuery**.



**Note:** The **Welcome to BigQuery in the Cloud Console** message box may appear, providing links to the quickstart guide and the release notes for UI updates. Click **Done** to proceed.

1. Expand the list of datasets by clicking the drop-down arrow next to the Project ID.

**Note:** You may need to select a project first. To do so, click **Select a project** in the Google Cloud console title bar, then select the project link from the **Select a project** dialog.

If you get a pop up, click **Ok**.

1. Find a dataset named **thelook\_gcda**, and click the drop-down arrow next to it.
2. Select the table named **shopping\_cart**.
3. Click the **Query** button to open the Query Editor, and select **In a new tab**. A pre populated **Untitled** tab opens.
4. Replace the pre populated data by copying and pasting the following query in the **Query Editor**:Copied!

```
SELECT *
FROM `thelook_gcda.shopping_cart`
ORDER BY created_at DESC
LIMIT 10;
```

content\_copy

This query displays the first 10 rows of the **shopping\_cart** table.

1. Click **Run**.

The query results should display in table format below the Query Editor.

**Hint:** In the Query Editor you can click on the "+" icon to compose a new query.

It's a busy shopping month. Meredith notices multiple items being added to shoppers' carts. Which column displays the time a product was added to the shopping cart?  
product\_id  
check  
created\_at  
None of these options  
session\_id

**Submit**

1. Note the time of the most recent entry, which will be displayed at the top of the table in Row 1.
2. Click the **Run** button to run the query again.

Is the time of the most recently added row (the top row) the same after you run the query again? Yes, streaming inserts does not reflect in the original table. `checkNo`, it's newer, because data is continuously added to the table. None of these options  
Yes, because the query returns results based on batch load data.

### Submit

Data is continuously added to this table, so queries can be run on the latest data as soon as it is available. This changes the output of the query.

1. Click on the `shopping_cart` table in the Explorer area, then click on the **Details** tab in the query area. Examine the `shopping_cart` table properties in BigQuery. The section **Streaming buffer statistics** is an indication that data is being streamed into the table.

**Tip:** Refresh the page if it does not display right away. You may need to scroll through to navigate the statistics.

1. Now, examine the properties of the `orders` table. Click on the `orders` table in the Explorer area, then click on the **Details** tab in the query area.

Is there a streaming buffer associated with the `orders` table? Yes, streaming inserts are reflected in the table. `checkNo`, because the table is not connected to a streaming source. It's loaded as a batch of rows rather than through streaming. No, as no new data is being added to the table. None of these options

### Submit

1. Copy the following into the **Query Editor**:

```
SELECT *
FROM `thelook_gcda.orders`
ORDER BY created_at DESC
LIMIT 10;
```

Copied!

content\_copy

1. Click **Run** display the most recent rows on the `orders` table.
2. Click **Run** again to examine if there is a change in the result.

Does the most recent sale change? None of these options `checkNo`, the most recent sale will not change unless the next batch update is executed. The changes will reflect after a few minutes. Yes, the data has changed.

### Submit

1. Copy the following query into the **Query Editor**:

```
SELECT
  p.category,
  FORMAT_TIMESTAMP("%H:%M", sc.created_at) as added_at_minute,
  sum(sc.quantity) as sum_quantity
FROM
  `thelook_gcda.shopping_cart` sc
INNER JOIN
  `thelook_gcda.products` p
ON
  p.id = sc.product_id
WHERE
  p.category = 'Jeans'
  AND sc.created_at > timestamp_sub(current_timestamp(), INTERVAL 1 HOUR)

GROUP BY p.category, added_at_minute
ORDER BY added_at_minute DESC;
```

Copied!

content\_copy

This query displays the number of products from the 'Jeans' product category that were added to the `shopping_cart` table each minute within the last hour.

1. Run the query at least three times, waiting about ten seconds between runs, and observe the results.

This query is an example of a data source for a dashboard. Because the query is based on the `shopping_cart` table, which is constantly updated through a streaming source, the dashboard will get fresh data every time the query runs.

Click **Check my progress** to verify that you have completed this task correctly.

You have successfully completed this task.

Determine batch versus streaming sources

### Check my progress

*You have successfully completed this task.*

**Now**, let's learn more about this query and its main components.

1. Locate the names of the tables used in this query.

The query joins

the **thelook\_gcda.shopping\_cart** and **thelook\_gcda.products** tables using the `product_id` column. This ensures that only rows that match the product id on both tables are included in the results.

What columns are referenced in this query?  
The quantity column from the shopping cart table  
The created\_at column from the shopping cart table  
The category column from the products table  
 All of these options

**Submit**

1. Identify the type of join used in the query:

```
FROM
`thelook_gcda.shopping_cart` sc
INNER JOIN
`thelook_gcda.products` p
ON
p.id = sc.product_id;
```

An `INNER JOIN` is used in this query. A shopping cart row with a value in the `product_id` that does not exist in the products table will not be included in the results, and therefore will not be counted.

Because Meredith asked for minute-by-minute information, the query formats the `created_at` column as HH:MM, for example 10:15 for a quarter past ten in the morning.

1. Identify the following statement in the `SELECT` portion of the query:

```
FORMAT_TIMESTAMP("%H:%M", sc.created_at) AS added_at_minute,
```

Formatting dates and times is a very common task for a cloud data analyst. You can find more information and examples on how to use the built-in `FORMAT_TIMESTAMP` function in BigQuery's [Timestamp functions documentation](#).

1. Locate the `WHERE` statement in this query.

The `WHERE` statement in this query filters shopping cart items by **Product Category** and by the time it was created:

```
WHERE
p.category = 'Jeans'
AND sc.created_at > timestamp_sub(current_timestamp(), INTERVAL 1 HOUR);
```

Since Meredith is only interested in Jeans, the following part of the query removes products not in that category with the first part of the `WHERE` clause:

```
p.category = 'Jeans'
```

The shopping cart table is very active and can be quite large, so Meredith also asked to see data only for the last hour. To accomplish this, the `WHERE` clause filters rows using the `created_at` column and uses two built in functions, one to get the date and time when the query runs (`current_timestamp`) and one to calculate the time for one hour ago (`timestamp_add`). The query will only return rows that were created before **now minus one hour**. **Now** is determined by the `current_timestamp` function. **Minus one hour** is done by subtracting **1** hour from the current time.

**Note:** *The time displayed in query results is in the Universal Time Coordinated (UTC) time zone, and it might not align with the time at your location. Storing time in UTC is a common practice in global companies.*

For more information, refer to the [Timestamp functions documentation](#).

1. Locate the following line in the query:

```
GROUP BY p.category, added_at_minute;
```

The following line groups the results by minute and adds up the quantities added for each minute because we used the `SUM()` function in our `SELECT` statement. Since you need to list the **Product Category** in the results, you must also group by **Product Category**.

Notice that you are using the alias of the `added_at_minute` column that was defined in the `SELECT` portion of the query.

1. Locate the `ORDER BY` clause in the query:

```
ORDER BY added_at_minute DESC;
```

The `ORDER BY` clause populates the results with the most recent available data, by the minute, at the top.

## Conclusion

Great work!

You were able to locate the information Meredith needed to stay on top of customer buying trends, which will help them make timely business decisions about pricing and stock.

You did this by first determining if the table was using batch or streaming processing, then exploring and running a query to populate a table with data about the number of jeans that were added to shopping carts in the last hour, by minute.

You also learned about the importance of timely data access and the difference between batch processing and streaming processing.

By observing data loading in BigQuery tables using both methods, you developed the skills to choose the appropriate approach for data processing.

## End your lab

Before you **end the lab**, make sure you're satisfied that you've completed all the tasks. When you're ready, click **End Lab** and then click **Submit**.

Ending the lab will remove your access to the lab environment, and you won't be able to access the work you've completed in it again.

Copyright 2024 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.

## ▼ Key Components of data Organization

## ▼ Denormalized Data

Denormalization is a data management technique that involves organizing data in a way that allows for faster retrieval and reduced complexity, typically at the expense of increased storage needs and potential inconsistencies. Unlike normalized data, which separates related fields into distinct tables to avoid duplication and maintain defined relationships, denormalized data consolidates information by storing repeated details across one or more tables.

### Key Characteristics:

- **Normalization** organizes related data into separate tables, making it easier to avoid redundancy and inconsistencies while enabling straightforward updates. For instance, employee data can be split into one table containing employee IDs and names, and another table for department information, linked via manager IDs.
- **Denormalization** simplifies data access by consolidating related information into fewer tables. This method enhances query performance but may lead to issues such as duplicate data and increased storage requirements.

| Data format  | Advantages   |
|--------------|--|
| Normalized   | <ul style="list-style-type: none"><li>• Avoids duplicate data and inconsistencies</li><li>• Quickly applies updates</li><li>• Optimizes memory to promote performance</li><li>• Maintains integrity</li></ul>  |
| Denormalized | <ul style="list-style-type: none"><li>• Improves execution time for queries</li><li>• Reduces number of tables to cut down on storage resources</li><li>• Is an alternative to expensive table JOINs</li><li>• Enhances scalability for changing data needs</li><li>• Reduces overall complexity</li></ul> |

### Common Denormalization Techniques:

1. **Adding Duplicate Columns:** Including duplicate columns in tables reduces the need for complex joins, making it easier for data analysts to access required information.
2. **Splitting Tables:** Creating smaller tables that contain only the rows or columns necessary for specific applications.
3. **Mirroring Tables:** Making copies of existing tables for easier reading and access.
4. **Summary Tables:** Generating pre-aggregated data tables, which can contain totals, counts, or averages, to streamline reporting.

### **Practical Example:**

In a sports company, a data analyst named Juliana must adapt to a new business objective that requires tracking customer activity times. While the company's data is initially stored in normalized tables, to efficiently retrieve customer names, emails, and activity times simultaneously, the team might consider transitioning to a denormalized structure. The decision to denormalize depends on various factors, including business goals, application features, and the urgency of data retrieval.

In conclusion, while denormalized data facilitates quick access to information, it is crucial to weigh the benefits against the drawbacks, particularly regarding data integrity and storage implications.

## **▼ Data Governance for Effective Data Management**

Data governance is a structured process designed to ensure the formal management of a company's data, promoting data accuracy, reliability, and security. In a world saturated with data, effective governance helps organizations maintain control and leverage data as a strategic asset. The key components of data governance include data policies and standards, data quality management, data privacy and security, and data stewardship.

### **Core Elements of Data Governance:**

- 1. Data Policies and Standards:** These define rules and guidelines for data management, ensuring consistent handling, usage, and sharing of data across an organization.
- 2. Data Quality Management:** Focuses on maintaining the accuracy, completeness, and reliability of data through processes like data cleaning, validation, and monitoring.
- 3. Data Privacy and Security:** Involves implementing security protocols, access controls, and ensuring compliance with data protection laws.
- 4. Data Stewardship and Ownership:** Assigns responsibility to specific individuals or teams for overseeing and managing data governance processes.

### **Key Goals of Data Governance:**

- **Data Integrity and Accuracy:** Ensuring data remains trustworthy and reliable throughout its lifecycle through validation and quality controls.
- **Compliance:** Aligning with data protection regulations to safeguard privacy and meet legal obligations.
- **Accessibility and Usability:** Making data more accessible to authorized users through classification, access controls, and metadata management.

- **Informed Decision-Making:** Establishing a foundation for making reliable, data-driven decisions by enhancing the quality and availability of data.
- **Security:** Reducing the risk of data breaches and privacy violations through robust governance protocols.
- **Efficiency:** Streamlining data management processes, reducing redundancies, and improving organizational effectiveness.

#### **Steps for Implementing Data Governance:**

1. **Set Clear Goals and Objectives:** Define what the governance initiative aims to achieve.
2. **Gain Leadership Support:** Secure backing from top-level executives to ensure the initiative's success.
3. **Form a Cross-Departmental Team:** Involve representatives from IT, legal, finance, and operations to ensure broad-based governance.
4. **Define Roles and Responsibilities:** Clearly outline the roles within the data governance team.
5. **Create a Governance Plan:** Develop policies, procedures, and standards for managing data.
6. **Invest in Tools and Technologies:** Use automation tools to streamline data management processes.
7. **Train the Team:** Ensure the team is well-versed in the tools and best practices.
8. **Monitor Progress:** Use key performance indicators (KPIs) to track the effectiveness of governance efforts.
9. **Promote a Data-Driven Culture:** Foster an environment where data is valued and insights are shared across the organization.
10. **Evolve with the Organization:** Continuously adapt governance processes and technologies to the organization's growth.

By establishing strong data governance, organizations can build trust, ensure compliance, enhance decision-making, and increase operational efficiency. Proper governance allows businesses to treat data as a vital asset while minimizing risks and aligning with their long-term goals.

#### **▼ Introduction to Master Data Management (MDM)**

Master Data Management (MDM) is a strategic framework used by organizations to ensure that critical data assets are consistently accurate, reliable, and accessible across the enterprise. This concept is essential for creating a **single source of**

**truth** that provides a unified and up-to-date view of core entities such as users, products, locations, and employees.

#### **Key Components of MDM:**

1. **Data Governance:** This involves establishing formal policies, standards, and processes to manage and protect data assets. It defines roles and responsibilities to ensure data integrity, security, and compliance.
2. **Data Integration:** It focuses on harmonizing data across various systems and departments. By ensuring seamless data flow, this component guarantees that every system uses consistent, up-to-date information from a single source of truth.
3. **Data Quality Management:** This is the process of ensuring the accuracy, completeness, and consistency of master data. It includes data cleaning, validation, and enrichment efforts to maintain high data quality standards.
4. **Data Stewardship:** This involves assigning specific individuals or teams to be responsible for data management, ensuring accountability for data accuracy, issue resolution, and adherence to governance practices.

#### **Business Benefits of MDM:**

- **Improved Data Quality:** MDM implements standardized rules and controls, eliminating duplicate or conflicting data and ensuring consistency across systems.
- **Informed Decision-Making:** MDM provides a comprehensive and reliable view of data, enabling data-driven decisions that support business growth and competitiveness.
- **Single Source of Truth:** By synchronizing data across systems, MDM eliminates data silos and ensures that all users access the most accurate and up-to-date information.
- **Operational Efficiency:** MDM reduces redundancies and errors in data management, saving time, effort, and money. It streamlines processes by eliminating the need for costly reconciliation, rework, and manual data entry.

In conclusion, implementing MDM not only enhances data quality and operational efficiency but also drives better decision-making and ensures the overall success of an organization's data management strategy. MDM creates a reliable foundation for managing critical business data, leading to improved performance and a stronger competitive edge.

### **▼ Introduction to Data Catalogs**

In today's data-driven world, every interaction—whether a click, a social media post, or a website visit—leaves a trail of data. Organizations collect vast amounts

of this data, which can quickly become overwhelming. However, data catalogs are an essential tool that helps streamline and organize these data assets, enabling data analysts to unlock powerful insights efficiently.

## What is a Data Catalog?

A **data catalog** is a centralized inventory that acts as a comprehensive guide to an organization's data assets. Just like a compass, it helps users navigate through the vast landscape of data to locate and use the right datasets when needed. It offers visibility and accessibility, laying the foundation for efficient data management.

### Key Features:

- Provides a centralized view of all available data assets.
- Facilitates easy discovery, understanding, and usage of data.
- Improves searchability, enabling users to find the right data effortlessly.

## Use Case: Sports Data

Imagine you're a data analyst for a sports team. The team collects a wide variety of data, including:

- **Structured data:** Player statistics over the team's history.
- **Unstructured data:** News articles, photos, videos, and social media content.
- **Reports:** Payroll and player performance reports.
- **Dashboards & Visualizations:** Metrics focusing on player performance.
- **Automations & Machine Learning Models:** Predictive tools for player performance.

With such an incredible volume of data, finding the right dataset for a specific player can be a challenging task. But with a data catalog, you can search all of these assets in one place, making it easier to compile and analyze player data.

## Key Elements of a Data Catalog

### 1. Metadata Management

Metadata, or "data about data," is critical in a data catalog. It helps to describe, organize, and enhance data value. For example, metadata allows users to understand a dataset's content, structure, and usage without directly accessing the raw data itself.

### 2. Data Lineage

Data catalogs track the origin, transformations, and history of each dataset. This is known as **data lineage**. It ensures data reliability by allowing users to follow its journey and trace any modifications made over time.

### 3. Data Governance

Data governance is the process of managing data policies and ensuring data security and privacy. Data catalogs play a crucial role by enforcing guidelines that regulate how data is accessed and used across the organization, ensuring compliance with relevant data regulations.

### 4. Collaboration

A well-implemented data catalog acts as a shared platform for teams to collaborate on data assets. Analysts, data scientists, and business stakeholders can work together, sharing insights and contributing to a more data-driven culture.

## Benefits of a Data Catalog

- **Enhanced Data Understanding:** By providing comprehensive metadata, data catalogs enable deeper insights, improving the accuracy of analysis and decision-making.
- **Increased Efficiency:** With organized data and metadata management, users can quickly locate and access key datasets, reducing time spent on data preparation and boosting productivity.
- **Improved Governance:** Data catalogs enforce policies that maintain data integrity, privacy, and security, helping organizations meet compliance requirements.
- **Encourages Innovation:** By making data more accessible, data catalogs foster a culture of innovation and encourage data literacy across the organization.

Data catalogs are an invaluable tool for any organization dealing with large volumes of data. By centralizing and organizing data assets, data catalogs allow teams to make more informed decisions, ensure governance, and boost collaboration. In the evolving landscape of cloud data, data catalogs will continue to play a pivotal role in helping organizations stay data-driven and innovative.

Keep exploring the world of data catalogs and see where your data journey takes you!

## ▼ Technical and Business Metadata

Metadata is a critical component in data management, providing essential information about the data we work with. Whether we realize it or not, metadata

plays a key role in our everyday digital activities. For instance, every web search you perform returns results built around metadata, such as page titles, descriptions, keywords, and the authors or sources of the content. This type of "data about data" helps describe, organize, and contextualize information, making it easier to understand and use.

In the realm of data management, metadata helps data professionals search, analyze, and derive insights from complex datasets. It ensures that data is both discoverable and meaningful. There are two primary types of metadata that serve distinct but complementary roles: **technical metadata** and **business metadata**.

## What is Metadata?

Metadata provides the crucial details that describe data, including information about its structure, format, origin, and meaning. In simpler terms, metadata answers the "what," "when," "where," and "how" about a dataset. Without metadata, finding and using the right data would be like searching for a needle in a haystack.

### Common attributes of metadata include:

- **Structure:** How the data is organized (e.g., tables, columns).
- **Format:** The type of data (e.g., CSV, JSON, SQL).
- **Origin:** Where the data came from.
- **Quality:** How reliable and accurate the data is.
- **Meaning:** What the data represents in a business context.

## Types of Metadata

Metadata can be broadly classified into two key types: **technical metadata** and **business metadata**. Both types provide different kinds of insights and context, yet they work together to enable effective data management.

## Technical Metadata

### Definition:

Technical metadata is all about the nuts and bolts of data. It focuses on the detailed technical aspects such as file formats, structure, and the lineage of the data (i.e., where the data has traveled throughout a system and how it has changed over time).

### Key Features:

- **File formats** (e.g., CSV, XML, JSON).
- **Data structure** (e.g., columns in a table or hierarchy in a dataset).
- **Source and lineage:** Tracking the data's origins and the transformations it has undergone.
- **Interoperability:** Ensuring the data can be exchanged between different systems and applications.

### **Importance:**

Technical metadata lays the foundation for data flow, accessibility, and integration across various systems. It ensures that data can be used smoothly across platforms, allowing for seamless data exchange and integration. This type of metadata is crucial for IT professionals, as it enables data compatibility and smooth functioning within technical environments.

## **Business Metadata**

### **Definition:**

While technical metadata explains the "how," business metadata focuses on the "why." It provides the business context, meaning, and rules associated with data, helping non-technical users understand its significance in achieving business objectives.

### **Key Features:**

- **Context and meaning:** Explains what the data represents within the business.
- **Business rules:** Guidelines on how the data should be used or interpreted.
- **Data ownership:** Identifying who is responsible for managing the data.
- **Relationships:** How the data is connected to other business processes or datasets.

### **Importance:**

Business metadata helps bridge the gap between technical complexity and business needs. It provides context for data analysts and decision-makers to understand how the data relates to real-world business goals, making data more accessible to non-technical stakeholders. Business metadata enables data-driven decision-making by ensuring that everyone in the organization can understand the data's relevance.

## Key Differences Between Technical and Business Metadata

While both types of metadata are essential for effective data management, it's important to understand the distinctions:

| Technical Metadata   | Business Metadata                             |
|--|---|
| Focuses on technical aspects (formats, structure, lineage) | Focuses on business context and meaning       |
| Used primarily by IT and data professionals                | Used by business analysts and decision-makers |
| Ensures interoperability and data exchange                 | Ensures data relevance to business objectives |
| Tracks data transformations over time                      | Defines business rules and data ownership     |

## Conclusion

Both technical and business metadata play crucial roles in data management.

**Technical metadata** focuses on the structural and technical details that allow data to flow smoothly between systems, while **business metadata** provides the context that enables organizations to make informed, data-driven decisions. Together, they form a comprehensive framework for understanding, organizing, and leveraging data effectively.

Understanding the distinctions and interplay between these two types of metadata is essential for anyone working in data management. As you advance in your cloud data career, mastering the use of metadata will be a key skill in navigating complex datasets and driving valuable insights.

## ▼ Overview of Data Lakehouse Architecture

### Introduction

Data lakehouse architecture combines the strengths of both data warehouses and data lakes, allowing organizations to manage vast amounts of data effectively. Understanding this architecture is crucial for data professionals as it impacts data organization and utilization.

#### 1. Data Warehouses:

- The concept of data warehouses emerged to provide a structured environment for storing and organizing data.

- Data warehouses are characterized by a centralized area where all data is routed for storage and analysis.
- The basic architecture includes data sources, the Extract, Transform, Load (ETL) process, and visualization or reporting tools.

## 2. Data Lakes:

- With the exponential growth of data in speed, volume, and variety, traditional data warehouses faced limitations.
- Data lakes emerged as an alternative, designed to store large amounts of raw, unstructured data in its original format until needed.
- They use cost-effective storage systems and allow for easier data ingestion, but maintaining quality and consistency becomes challenging.

# Data Lakehouse Architecture

## 1. Definition:

- A data lakehouse is a hybrid architecture that merges the features of data lakes and data warehouses.
- It maintains the flexibility of a data lake while incorporating structured elements and organization similar to a data warehouse, ensuring data quality and consistency.

## 2. Components:

- **Data Ingestion:** Data is gathered from various sources, similar to both data lakes and warehouses.
- **ETL Process:** Data undergoes transformation processes before storage or analysis.
- **Storage:** Data is stored in its native format, allowing for both structured and unstructured data management.
- **Data Processing:** Data can be analyzed, visualized, or used in machine learning models directly from the lakehouse.

# Advantages of Data Lakehouse Architecture

## 1. Scalability:

- Supports the storage and processing of vast amounts of data without needing additional hardware or software for transformation.

## 2. Flexibility:

- Enables organizations to leverage various data tools and technologies without compatibility issues.

### **3. Cost Efficiency:**

- Reduces storage and processing costs by utilizing cost-effective storage solutions.

### **4. Dual-Tier Architecture:**

- Organizations needing structured data can implement a two-tier architecture that incorporates both a data warehouse and a lakehouse.

## **Conclusion**

Data lakehouses empower organizations to store, process, and analyze data more efficiently. By understanding data lakehouse architecture, data professionals can enhance their data management skills and contribute to building effective data solutions in their organizations.

## **▼ Steps to Find Data**

### **▼ Data Lineage and Traceability**

Humans are naturally curious about the origins or sources of things. This curiosity drives us to investigate various aspects of our lives, whether it's to appreciate cultural traditions or to evaluate the ethical implications of the products we consume. In the data world, understanding the origins of data is equally crucial.

## **Importance of Data Lineage**

Data lineage refers to tracking the journey of data from its source through its lifecycle, highlighting transformations and movements along the way. This understanding is vital for ensuring data integrity and enabling data professionals to share insights confidently.

## **Defining Data Sources**

A data source is where data is generated. These can take many forms, including:

- **Databases:** Hold structured sets of data and store and manage large volumes of information.
- **Application Programming Interfaces (APIs):** Tools used to gather data from external systems, such as social media.
- **Files:** Various formats, such as Excel, CSV, or JSON, that allow data import/export between systems.
- **Streaming Platforms:** Sources that provide a continuous flow of live data for near real-time analysis, such as monitoring user interactions on a website.

## Challenges with Data Sources

The diversity of data sources can lead to challenges concerning compatibility, integration, and comprehension. Each data type necessitates a unique approach, and understanding their applications is essential for effective use.

## Tracking Data Lineage

Tracking lineage involves following data from its origin to its current state. This process helps to reveal where the data has been and how it may have changed over time. Determining data lineage can enhance the context surrounding the data, enable traceability, and support audits of its reliability and accuracy.

## Benefits of Data Lineage

- **Improved Decision-Making:** Data professionals can ask better questions and make informed choices.
- **Authenticity and Quality Assurance:** Clear understanding of data authenticity, quality, and integrity is crucial for organizations.
- **Control Over Data Assets:** Data lineage assists organizations in maintaining oversight of their data resources.

## Data Governance and Compliance

Data tracing plays a significant role in data governance, compliance, and regulatory requirements. Similar to checking food labels or researching companies before purchases, data lineage and traceability foster transparency, accountability, and trust.

## Conclusion

As data significantly influences our world, it is essential for cloud data professionals to analyze and verify data rigorously. Working with trustworthy data enables businesses to achieve better outcomes, highlighting the incredible role of data professionals in today's data-driven landscape.

## ▼ Introduction to Analytics Hub

### What is Analytics Hub?

Google Analytics Hub acts as a data exchange and a "library" of both internal and external data assets. In this document, we will explore Analytics Hub's architecture, user tasks, and the benefits it offers to data professionals.

### Functionality of Analytics Hub

Just like a library catalog, Analytics Hub organizes and secures data, enabling users to locate the datasets they need for analytics projects. It serves as a connector between data producers (publishers) and data consumers (subscribers), facilitating safe and private data sharing among organizations.

## Architecture of Analytics Hub

Analytics Hub is built on a publish-and-subscribe model based on BigQuery's datasets:

- **Publishers:** Data producers who make their datasets available.
- **Subscribers:** Data consumers who access these datasets.

This architecture separates compute and storage, allowing data producers to share datasets without duplicating them. Producers pay only for storage space, while consumers incur costs only when they access or query the shared data.

## Workflow of Data Sharing

1. **Dataset Sharing:** Data publishers identify datasets they wish to share in BigQuery and create a listing in Analytics Hub.
2. **Subscription Process:** After publishing the listings, subscribers can browse Analytics Hub and subscribe to the datasets of interest. A read-only link to the dataset is generated in the subscriber's BigQuery project, enabling data querying.

## Key Data Management Features

Analytics Hub offers several essential data management features:

1. **Shared Datasets:** Collections of data, tables, and views shared by data publishers in BigQuery.
2. **Linked Datasets:** Subscribers receive a version of the dataset that is read-only, ensuring the dataset remains unchanged and alleviating storage costs.
3. **Listings:** Each piece of data is uniquely identified, including links to the datasets, brief descriptions, and related documentation.
4. **Data Exchanges:** Data exchanges can be open to all users or restricted to specific users, with exchange administrators managing access permissions. By default, exchanges are private.

## User Roles in Analytics Hub

There are four main types of users in Analytics Hub:

1. **Publisher:** Generates income through instant data sharing and builds a catalog of data sources for analysis while managing access permissions.
2. **Subscriber:** Merges shared data with existing datasets and utilizes BigQuery's built-in tools for analysis.
3. **Viewer:** Browses datasets and requests permission to use shared data.
4. **Administrator:** Creates data exchanges and grants access permissions to both publishers and subscribers.

## Limitations of Analytics Hub

Data publishers should be aware of certain limitations when using Analytics Hub:

1. If a publisher creates a listing for a shared dataset encrypted with a customer-managed encryption key, subscribers will lack the cloud key to access it.
2. There is a limit of 1,000 linked datasets to a shared dataset.
3. Datasets with unsupported resources cannot be shared when creating a listing (e.g., not all routines are supported in shared datasets).

## Conclusion

Analytics Hub effectively connects those who create datasets with those who need them, functioning like a library that offers a wealth of well-managed and easily accessible datasets. As a cloud data professional, leveraging Analytics Hub facilitates the sharing and utilization of data among organizations, enhancing analytical capabilities and outcomes.

## ▼ Data Discovery, Curation, and Unification

In today's data-driven world, data is seen as the "currency of business success." However, the key challenge is distinguishing valuable insights from irrelevant data. Effective data management requires three core processes: data discovery, curation, and unification.

### 1. Data Discovery

Data discovery involves identifying and understanding data assets, as well as recognizing patterns and relationships within a dataset. It is essential to know what a business aims to achieve to guide data exploration and ask the right questions.

### 2. Data Curation

This process focuses on selecting, organizing, and maintaining high-quality data. Curation ensures data is well-preserved for future use and readily available for decision-making. By building a reliable data foundation, it enhances the analytical process.

### **3. Data Unification**

Data unification integrates data from various sources into a unified view, overcoming challenges posed by different formats and locations. Unification facilitates comprehensive analysis and helps analysts work with a complete dataset, leading to better insights.

#### **Cloud Benefits**

Leveraging cloud technology enhances these processes by offering:

- **Scalability:** Adapt to any dataset size.
- **Accessibility:** Seamless remote access to tools.
- **Automation:** Automates data curation and transformation, improving efficiency.
- **Security:** Cloud providers offer encryption and compliance frameworks to protect data.
- **Cost-Savings:** Cloud solutions reduce operational costs, freeing resources for further data discovery and analytics efforts.

By using the cloud, businesses can streamline data management, improve collaboration, and unlock valuable insights efficiently.

## **▼ Benefits of Using Dataplex**

Dataplex is a powerful, centralized data catalog designed to enhance data discovery, unification, and management, offering several key benefits:

### **1. Centralized Data Management**

Dataplex provides a unified platform where users can explore, catalog, and manage data assets across complex environments. This centralized approach simplifies data source identification, helping users easily navigate through vast data ecosystems.

### **2. Enhanced Data Discovery**

With integrated search capabilities, Dataplex allows users to find datasets quickly and efficiently, even in large-scale environments like BigQuery. Users can search for specific datasets or filter results by criteria, making data discovery seamless.

### **3. Contextual Data Understanding**

By offering metadata insights like data structure, schema, and lineage, Dataplex enables users to better understand the context of datasets. This is crucial for ensuring data reliability and accuracy in analysis.

### **4. Automation & Integration with Google Cloud**

Dataplex automatically scans data sources within Google Cloud, allowing instant access to various datasets. This automation reduces manual effort and accelerates data management processes.

## 5. Scalability & Performance

Integrated with BigQuery, Dataplex offers the ability to manage massive datasets with fast query performance and scalability. This makes it ideal for organizations dealing with large volumes of data.

In summary, Dataplex enhances data discovery, provides centralized control, and improves data management efficiency, making it a valuable tool for any data professional.

# ▼ Techniques to Access Data

## ▼ Methods for Defining BigQuery Table Schemas

Understanding table schemas is crucial for data professionals as they provide clarity and structure to how data is organized. Here are three primary methods to create table schemas in BigQuery:

### 1. Manual Schema Definition via Google Cloud Console

- When loading data into a table, you can manually define the schema by specifying each column's name and data type.
- To do this, use the "Add field" option in the console and input the field name, type, and mode.
- It's important to know the data structure beforehand or to have a plan for the columns if starting with an empty table.

### 2. Schema Auto-Detection

- BigQuery offers a feature that automatically infers the schema based on existing data formats like CSV, JSON, or Google Sheets.
- To use this feature, select the "Auto-detect schema" option while creating a table.
- BigQuery samples a random file from the data source and scans up to 500 rows to determine the appropriate data types for each column.
- It's advisable to verify the assigned data types in the Google Cloud console or command line for accuracy.

### 3. Using a JSON Schema File

- A JSON schema file can be used to specify the schema, including column names, modes, data types, and descriptions.

- This method is applicable when creating a table from the command line but not from the console.
- To create a table from a JSON file, select “Google Cloud Storage” in the BigQuery Explorer, choose the JSON file, and set the file format to JSONL.
- You can also opt for auto-detection of the schema during this process.

## Example Scenario

In a case where a commercial real estate agency compiles client surveys, the data team may initially use schema auto-detection to quickly ingest the data. However, if they require more control over column names, they can recreate the schema using the Google Cloud console. This flexibility allows for better understanding and analysis of agent performance, ultimately enhancing client experience.

By mastering these methods, data professionals can structure their data effectively, making it more manageable and insightful.

## ▼ Basic SQL Commands for Querying Data

Welcome to the introduction to basic SQL commands! Understanding SQL is essential for data professionals, as it allows you to interact with databases effectively. Here's a breakdown of fundamental SQL commands and concepts:

### 1. SQL Dialects

- **Definition:** SQL dialects are specific versions of SQL tailored to different database systems. While they share a core structure, there are minor variations in syntax and commands.
- **Importance:** Learning the basics of SQL provides a solid foundation for working with any database.

### 2. Key SQL Commands

As a data analyst, there are four essential commands you need to know:

#### a. Retrieving Information

- **Commands:** `SELECT`, `FROM`, and `WHERE`
  - `SELECT`: Specifies the data you want to retrieve (individual columns or all columns).
  - `FROM`: Identifies the table from which to pull data.
  - `WHERE`: Adds conditions to filter the data returned, such as retrieving only rows with specific values.

**Example:**

```
SELECT column1, column2  
FROM table_name  
WHERE condition;
```

## b. Selecting Columns

- Use the `SELECT` and `FROM` statements together to specify which columns to retrieve from a table.

### Example:

```
SELECT CustomerID, CustomerName  
FROM Customers;
```

## c. Sorting Data

- **Commands:** `SELECT`, `FROM`, and `ORDER BY`
  - Use `ORDER BY` to specify which column to sort by. By default, it sorts in ascending order. You can add `DESC` to sort in descending order.

### Example:

```
SELECT column1, column2  
FROM table_name  
ORDER BY column1 ASC; -- or DESC for descending
```

## d. Creating a Table

- Use the `CREATE TABLE` statement to define a new table and its schema.
- Specify the column names and their data types.

### Example:

```
CREATE TABLE Customers.CustomerNames (  
    CustomerID INT64,      -- Numeric type for unique identifier  
    CustomerName STRING    -- String type for names  
);
```

## 3. Common Data Types

- **INT64:** A numeric data type suitable for storing large integers (like identifiers).

- **STRING**: Used for storing text data (like names and addresses).
- Other common data types include **FLOAT**, **BOOLEAN**, etc.

## Example Scenario

Let's say you want to create a table named **CustomerNames** in the **Customers** dataset:

- The **CustomerID** will be an **INT64** for unique identification.
- The **CustomerName** will be a **STRING** for storing names.

Running the provided **CREATE TABLE** command will create an empty table with the specified structure, ready for data entry.

## Conclusion

By mastering these basic SQL commands, you will lay the groundwork for successful data querying and manipulation. As you gain experience with different databases, you will become proficient in navigating SQL dialects and contributing effectively to data analysis projects. Happy querying!

## ▼ Steps and Models for Accessing Data with Machine Learning

Data holds the key to transforming industries and shaping the marketplaces of the future. When combined with the exciting power of machine learning (ML), it opens a whole new world of insights and groundbreaking advancements. However, traditional methods of data analysis, like statistics, often limit the potential of ML.

## Defining Machine Learning

Machine learning involves the use and development of algorithms and statistical models to teach computer systems how to analyze and discover patterns in data. These ML algorithms rely on vast amounts of high-quality data to make accurate predictions and support smart decision-making.

## The Role of Cloud Computing

Accessing the right data at the right time is crucial for the success of ML projects, and this is where cloud computing becomes a game-changer.

## Real-World Example: Predicting Crop Yields

Let's consider an example involving a data analyst named Zane. Zane's organization wants to predict annual farming yields in a specific agricultural community. However, the organization has only been tracking crop yields from public datasets that include information about communities across the country—lacking specific data about local crop yields.

Zane plans to create an ML model for the prediction, but he needs more data to make it work. Here, the cloud comes to the rescue! Zane can request datasets from various organizations, and by ingesting this data—such as local farmers' datasets and national weather data—he can start training the model.

**Pro Tip:** Datasets from private organizations often require permission and coordination to access. After obtaining permission, you'll need a secure method for data transfer.

## Steps for Accessing Data via Cloud Services

1. **Choosing a Cloud Service Provider:** The first step is selecting the right cloud service provider based on your organization's needs.
2. **Setting Up Credentials and Permissions:** Ensure you have the proper credentials and permissions to access data securely.
3. **Establishing Connections:** Next, establish a secure connection to external datasets.

## Methods for Data Access

To ingest data from external datasets, you need credentials—similar to logging into a personal account. Credentials typically include a username and password, and organizations may provide them to access and transfer data.

Common interfaces for transferring data include:

- **Application Programming Interfaces (APIs):** A protocol enabling one application to interact with another.
- **Software Development Kits (SDKs):** Tools that include code libraries to access specific datasets or data points.

In Zane's case, an API allows him to transfer data from national agriculture databases into his system directly, streamlining the process and providing a direct pathway to the ML model.

## Machine Learning Models

Once the data is ingested, you can select or create an ML model. Here are some common types:

1. **Classification Models:** These models classify items into predefined categories based on common characteristics. For instance, classifying animals as mammals or reptiles, or predicting whether a product is likely to sell.

2. **Clustering Models:** These group data points that are similar. The ML model determines how to group items based on their similarities. For example, identifying geographic areas with customers likely to purchase specific products.
3. **Linear Regression Models:** Linear regression estimates the relationship between a dependent variable and one or more independent variables. This model can be used to predict sales or rainfall.
4. **Ranking Models:** These models rank items by likelihood. For example, streaming services use ranking models to recommend music or movies based on user preferences.

## Zane's Crop Prediction Model

After collecting and analyzing the necessary datasets, Zane uses a **regression model** to predict the crop yield for the upcoming year. With access to the right data, Zane is now on track to build a robust model for crop prediction.

## The Power of the Cloud in Machine Learning

The cloud enables continuous access to diverse datasets and maximizes the power of ML. With cloud-based solutions, ML models can improve over time, leading to better predictions and smarter insights. By leveraging the cloud, you can unlock the full potential of data-driven discovery.

### ▼ Introduction to Machine Learning with Vertex AI and BigQuery

Machine learning (ML) is a powerful technology with endless possibilities. It can:

- Identify objects in photos
- Analyze medical images
- Understand and process natural language (for chatbots, voice assistants, and translations)
- Recommend products, services, or content (helping users find the right online store, movie, or playlist)

As a data professional, you may find opportunities to train and deploy ML models. This guide introduces **Vertex AI**, a machine learning platform, and explains how it works with **BigQuery** for data collection.

## What is Machine Learning?

Machine learning uses algorithms and statistical models to teach computers how to analyze data and discover patterns. It helps professionals:

- Make better decisions

- Automate processes
- Uncover valuable insights

## Vertex AI: Taking ML to the Next Level

Vertex AI is a comprehensive platform for developing, deploying, and managing ML models at scale. It supports **machine learning operations (MLOps)**, which helps manage and streamline ML workloads. This makes your data pipeline faster and more efficient, allowing you to focus on innovation.

### Real-World Example: Improving Shipping Time Predictions

Let's consider an example. You're a new data analyst working for an online store and marketplace. Your team is tasked with improving the accuracy of shipping time estimates for each product. In the past, your company built machine learning models and updated shipping times monthly, but this process was slow and often inaccurate.

**The solution:** Your team moves the on-premises data warehouse to **Google Cloud** and builds a scalable pipeline that ingests data in near real-time.

Once the data is ready for ML, the next step is to automate **hyperparameter tuning**. Hyperparameters control the learning process in machine learning models. Vertex AI simplifies this by automatically adjusting the hyperparameters to find the optimal values.

### Automating Models with Vertex AI

With hyperparameter tuning automated, the focus shifts to updating delivery time predictions. **Vertex AI** allows your ML models to automatically update these predictions as new data flows into the system. This ensures accuracy and saves time.

## Cloud Storage: BigQuery

A good option for moving your on-premises databases to the cloud is **BigQuery**. It's not only a powerful cloud data storage solution, but also has built-in **machine learning capabilities**.

In traditional databases, building ML models would require advanced programming knowledge and specialized tools. BigQuery makes it easier—allowing you to develop ML models using **SQL**. This simplifies the process for professionals who are familiar with SQL but not advanced ML programming.

### Solving the Sales Forecasting Problem

Once your team has solved the shipping time issue, the next task is generating a daily **sales forecast**. To do this, you can use BigQuery's built-in **linear regression model** to predict sales based on inputs like:

- Products
- Past sales by date and hour
- Holiday sales patterns

Linear regression estimates the relationship between a dependent variable (e.g., sales) and one or more independent variables. With BigQuery, you can input this data using SQL, build a model, and let BigQuery handle the calculations. The model improves over time as more data is added.

## The Power of Machine Learning

Machine learning is a game-changing technology that can solve a wide range of problems and answer many types of questions. Platforms like **Vertex AI** and **BigQuery** make it easier than ever to build, manage, and automate machine learning models. With these tools, you can streamline your workflows and focus on what matters most—innovating and delivering valuable insights.

### ▼ Overview of Google Colab

Google Colab is a powerful, cloud-based platform designed for collaborative data analysis and machine learning tasks. In this overview, we will explore how Colab works, its integration with Jupyter Notebook and Python, and its benefits for data professionals.

#### 1. What is Google Colab?

- **Definition:** Colab, short for Colaboratory, is a cloud-hosted version of Jupyter Notebook that allows users to write and execute Python code in a web browser without the need for any configuration.
- **Key Feature:** It provides free access to computing resources, including graphical processing units (GPUs) and tensor processing units (TPUs), which are ideal for data-heavy tasks like machine learning.

#### 2. The Role of Jupyter Notebook and Python

- **Jupyter Notebook:** An open-source platform that allows the creation and sharing of documents that contain code, equations, visualizations, and text. It supports various programming languages, with Python being the most popular.
- **Python:** A flexible, object-oriented programming language that is extensively used for data analysis and machine learning. Python's adaptability makes it the

primary language in Colab, and it can be run directly from the browser without needing a command-line interface.

### 3. Colab's Features and Capabilities

- **No Setup Required:** Users can start coding in Python right from their web browsers (Google Chrome, Firefox, Safari) without having to install any software.
- **Integration with Google Drive:** Colab notebooks are stored in Google Drive, making them easy to share and collaborate on. Collaborators can comment, edit, and even run the code.
- **Rich Text Support:** Colab supports not just executable code but also rich text, images, HTML, and LaTeX, which makes documentation and presentation more interactive.

### 4. Benefits for Data Analysis and Machine Learning

- **Free Access to GPUs/TPUs:** Colab provides free access to powerful computing resources, such as GPUs and TPUs, for up to 12 hours. This allows users to analyze large datasets without the limitations of their personal computers.
- **Pre-installed Libraries:** Colab comes with several pre-installed libraries like NumPy and Keras, reducing setup time for users.
- **Compatibility with External Data Sources:** Colab can connect to a wide range of data sources such as AWS S3, Google Cloud Platform (GCP), and SQL databases. Users can easily import data into Colab for analysis and machine learning.
- **Version Control:** Colab keeps track of every change, ensuring users can review and revert to previous versions of their notebooks.

### 5. Collaboration and Sharing

- **Real-time Collaboration:** Users can share their Colab notebooks with colleagues, allowing them to run the code and collaborate on the same project. Integration with GitHub allows easy transfer of code files between platforms.
- **Customizable Sharing Options:** When sharing a notebook, users can choose to omit the code output, allowing others to run the code themselves and see the results.

### 6. Limitations

- **Resource Limits:** Although Colab offers free resources, they are not guaranteed or unlimited. Users may experience fluctuating availability of

computational resources based on demand.

- **Code Sharing:** When sharing notebooks, all content, including code, output, and comments, is shared by default, but users can adjust settings to omit code cell output if desired.

## Conclusion

Google Colab is a dynamic and accessible platform for data analysts, machine learning programmers, and researchers. By combining the functionality of a Jupyter Notebook with Python's powerful data processing capabilities, Colab provides an efficient environment for collaborative data-driven projects, all through an internet browser.

## ▼ Essentials of Database Partitioning

In this overview, we will dive into the key concepts and benefits of database partitioning, a crucial technique for improving the efficiency and scalability of data systems.

### 1. What is Database Partitioning?

- **Definition:** Database partitioning involves dividing large datasets into smaller, more manageable segments, or partitions, which can be managed and accessed separately. This process occurs across servers or databases, enhancing the system's performance and scalability.
- **Difference from Table Partitioning:** Unlike table partitioning, which segments data within a single table, database partitioning distributes data across multiple systems, making it a powerful tool for handling large-scale data environments.

### 2. Benefits of Database Partitioning

Database partitioning offers several advantages, including:

- **Improved Scalability:** Partitioning allows data to be distributed across multiple servers, enabling the system to handle larger workloads more efficiently.
- **Enhanced Availability:** By spreading data across multiple partitions, database partitioning helps prevent a single point of failure. Built-in backup features further ensure that services remain available even in case of hardware or software failures.
- **Better Performance:** Partitioning optimizes performance by allowing queries to access only a specific segment of the data rather than the entire database, leading to faster query processing.

### 3. Key Database Partitioning Strategies

There are three primary strategies for partitioning data:

- **Horizontal Partitioning (Sharding):** In this method, data is divided into smaller segments, called shards, where each partition holds a unique subset of the overall dataset. For example, an online store may shard customer orders based on region or user groups, improving both scalability and performance.
- **Vertical Partitioning:** This involves dividing data by field, placing frequently used fields in one partition and less commonly accessed fields in another. For example, a sneaker inventory could have a partition containing commonly queried fields (e.g., name, color, price) and another partition with fields accessed less frequently (e.g., current stock, location).
- **Functional Partitioning:** Data is grouped according to its function or use in the system. For instance, sneakers could be partitioned based on function (e.g., running, training) or by pricing strategy. This approach enables specialized handling of data depending on its purpose.

## 4. Key Considerations for Partitioning in Active Systems

When incorporating partitioning into an existing system, there are important factors to consider:

- **Data Access Changes:** Partitioning may require rethinking how data is accessed in the system to ensure smooth integration.
- **Data Migration:** Partitioning an active system typically involves migrating large amounts of data, which must be carefully managed to minimize disruptions.
- **Maintaining System Availability:** During the migration, the system should remain operational so that users can continue their work without significant interruptions.

## 5. Parallel Processing and Application Requirements

- **Parallel Processing:** Partitioning enables parallel processing, which enhances query performance by allowing different parts of the data to be processed simultaneously. This is especially useful for large datasets and complex queries.
- **Application Requirements:** Consideration of application requirements is crucial when designing a partitioned system. Understanding how data will be queried and changed helps ensure the system remains responsive, reliable, and performs well.

## 6. Rebalancing Partitions

- **Rebalancing:** As traffic patterns change, it may become necessary to rebalance partitions to avoid uneven distribution of workloads. This can involve creating a new partitioning strategy and migrating data accordingly.

## Conclusion

Database partitioning is a powerful tool for improving query performance, scalability, and availability. By dividing data into smaller, manageable subsets, you can ensure faster and more efficient data analysis. Implementing an effective partitioning strategy will give your organization a significant advantage in managing its data systems!

## ▼ Methods for Partitioning Tables

Imagine searching through a pile of papers to find a specific document—it can be time-consuming. But if those papers were neatly organized into labeled sections, the task would be much easier. Similarly, in a database, partitioned tables help organize data into manageable sections, improving efficiency in both data management and queries.

### 1. What is Table Partitioning?

- **Definition:** Table partitioning divides large tables into smaller segments, called partitions. Unlike database partitioning, which manages multiple databases, table partitioning keeps the data within a single table but separates it into sections that can be queried more efficiently.
- **Benefits:** It reduces the amount of data a query needs to process, enhancing query performance and helping control costs. Additionally, the table's metadata can further optimize query performance by providing cost estimations before running the query.

### 2. When to Partition a Table?

Table partitioning can be helpful in the following situations:

- **Improving Query Performance:** Partitioning allows queries to scan only specific sections of the table, resulting in faster query times.
- **Handling Large Volumes of Data:** When data operations exceed expected volumes, partitioning limits the query to specific partition column values, preventing performance slowdowns.
- **Managing Data Expiration:** Partitioning is beneficial when there's a need to set an expiration schedule to delete data after a certain period automatically.
- **Selective Data Loading:** Data can be loaded into a specific partition without affecting other partitions.

- **Targeted Deletion:** Specific partitions can be deleted without scanning the entire table.

### 3. Partitioning Methods

There are three primary ways to partition a table:

- **Integer-Range Partitioning:** This method partitions a table based on ranges of values in an integer column. For example, you can set starting and ending values with specific intervals for range partitioning.
- **Time-Unit Column Partitioning:** In this method, partitioning is based on a date, timestamp, or datetime column. Data is automatically placed in the appropriate partition based on these time values, and you can set the partitions to hourly, daily, monthly, or yearly divisions depending on the column type.
- **Ingestion-Time Partitioning:** Here, BigQuery automatically assigns rows to partitions based on the time the data is ingested into the system. This is useful when managing data based on when it is added to the database.

### 4. Clustering in Tables

In addition to partitioning, **clustering** is another option for improving query performance.

- **Clustering Defined:** A clustered table is organized based on values within specific columns (called clustered columns). Data is sorted by these clustered columns, which speeds up queries that frequently access data from these columns.
- **Use Case Example:** Consider a sales table with columns like *Product*, *Date*, and *Region*. If the *Date* column is chosen as the clustered column, all rows with the same date will be grouped together. This arrangement makes querying more efficient since the data is already sorted in the desired order.
- **Combination of Partitioning and Clustering:** In some cases, you can combine partitioning and clustering. For instance, you can partition a table by date and cluster it by a specific field, such as region or product, to improve the performance of queries that filter by both date and that field.

## Conclusion

Whether using **partitioning** or **clustering**, both methods offer significant benefits in improving data management and query performance. These techniques, much like organizing a folder, help you quickly locate and retrieve the data you need, providing faster and more efficient access to large datasets.

## ▼ Strategies for Querying Partitioned Tables

Partition pruning in databases is similar to pruning a tree, where unnecessary branches are cut off to improve the tree's health and productivity. Similarly, **partition pruning** is the process of eliminating unnecessary data from consideration when running a query, making the data work more efficient and insightful.

## 1. What is Partition Pruning?

- **Definition:** Partition pruning eliminates irrelevant data by scanning only the partitions relevant to a query. It allows data professionals to filter the value of the partitioning column, telling BigQuery to scan only the partitions that match the filter and skip the rest.

## 2. Queries for Different Types of Partitioned Tables

Each type of partitioned table has its own pruning strategy:

### a) Time-Unit Column Partitioning

- **Example:** If a dataset is partitioned by a date column, you can apply a filter to exclude dates outside a specific range. For instance, a query might exclude dates before January 1, 2016.

### b) Ingestion-Time Partitioning

- In BigQuery, ingestion-time-partitioned tables come with a special column called **partition time**, which represents the UTC timestamp of when each row was imported, rounded to the nearest partition boundary (e.g., hourly, daily).
- **Partition Date:** For tables partitioned by the day, BigQuery includes a pseudo-column called **partition date**, which is derived from the partition time but shows only the date.
- **Example:** To prune partitions in these tables, you can filter the partition time or partition date columns. For example, you might scan partitions between January 1, 2016, and January 2, 2016.

```
SELECT * FROM dataset.table  
WHERE transaction_date >= '2016-01-01'
```

### c) Integer-Range Partitioning

- **Example:** For tables partitioned by an integer column (e.g., Customer ID), you can filter partitions based on the value ranges. If a table is partitioned into ranges of 10 for Customer IDs, querying the partitions for IDs between 30 and 50 will only scan the corresponding partitions (30, 40, and 50).

```
SELECT * FROM dataset.table  
WHERE customer_id BETWEEN 30 AND 50
```

#### d) Unpartitioned Partition

- Data written to BigQuery in near real time is stored temporarily in a partition called **unpartitioned**, which holds null values in the partition time and partition date columns. You can query this unpartitioned partition by filtering for null values in the partition time pseudo-column.

```
SELECT  
    column  
FROM dataset.table  
WHERE  
    _PARTITIONTIME IS NULL
```

### 3. Best Practices for Partition Pruning

To design effective partition pruning strategies, here are some best practices:

- **Use Constant Filter Expressions:** Always include a filter expression to limit the partitions being scanned. This reduces the amount of data processed, improving the efficiency of the query.
- **Isolate the Partition Column in Filters:** Ensure that filters are based on the partition column itself. Filters that require data from multiple fields won't prune partitions, leading to less efficient queries.
- **Require Partition Filters:** Enforce partition filters in queries to ensure that unnecessary partitions are always excluded. This guarantees that only relevant partitions are scanned, saving time and resources.

### Conclusion

Partition pruning enhances query performance on partitioned tables by eliminating irrelevant data, allowing analysts to focus on relevant partitions. Just as careful pruning helps a tree grow stronger, partition pruning makes data queries more efficient and effective.

## ▼ Key Processes and Benefits of Dataproc

Google Cloud's **Dataproc** is a fully managed service designed to run big data processing jobs efficiently. It integrates open-source data tools, like Hadoop and Spark, allowing organizations to streamline data workflows. Here's an overview of

the key processes and benefits of Dataproc for data professionals, particularly data analysts.

## 1. Key Processes in Dataproc

Dataproc enables a variety of data processing methods:

- **Batch Processing:** Dataproc can handle large volumes of data collected over time, processing it all at once. This is useful for tasks that don't require real-time data but need to process data in bulk.
- **Querying:** You can request data from a database via Dataproc, enabling efficient data extraction and analysis.
- **Streaming:** Dataproc can work with data as it's generated, enabling real-time analytics and data-driven decisions.
- **Machine Learning:** Dataproc supports the use of machine learning algorithms to analyze large datasets, discover patterns, and create predictive models.

## 2. Technologies and Tools

Dataproc uses popular open-source frameworks:

- **Hadoop:** Dataproc leverages Hadoop for distributed data processing, distributing tasks across clusters of computers, enabling the efficient handling of massive datasets.
- **Spark and PySpark:** Dataproc supports Spark, including PySpark, a Python API for Spark, allowing the creation of automated analyses and pipelines for both structured and semi-structured data.

## 3. Benefits of Dataproc

Dataproc offers multiple benefits to organizations:

- **Cost Efficiency:** One of Dataproc's key features is the ability to turn off clusters when they are not needed. This saves costs by ensuring that organizations only pay for the compute power they use.
- **Faster Data Processing:** By distributing workloads across multiple machines in a cluster, Dataproc speeds up data processing tasks, significantly reducing the time spent on manual effort.
- **Integration with BigQuery:** Dataproc integrates with BigQuery, Google Cloud's data warehousing service, providing analysts with a single platform for querying and analyzing their data. This simplifies workflows, removing the need to switch between different tools.

- **Scalability:** Clusters in Dataproc scale up and down based on demand. This means that resources are optimized to meet the needs of the current data processing job, reducing costs and improving efficiency.
- **Managed Service:** Dataproc is fully managed, which means data professionals don't need to worry about maintaining infrastructure or performing updates. Google Cloud handles all maintenance tasks, leaving analysts free to focus on extracting insights from the data.
- **High Availability:** Dataproc ensures that clusters and data processing jobs are highly available, allowing for the seamless ingestion of data multiple times per day without interruption.

## 4. Enhancing Productivity

By automating repetitive data processing tasks, data analysts can focus more on analyzing the data and uncovering valuable business insights, ultimately improving the organization's decision-making processes.

In conclusion, Dataproc simplifies data processing, reduces costs, and speeds up workflows, making it a powerful tool for any data-driven organization.

# ▼ 3- Data Transformation in the Cloud

## ▼ Introduction to data transformation in the cloud

### ▼ Stages of the Data Journey

In this lesson, we'll explore how data moves through the cloud using a process called the **data journey**. The data journey begins when you, as a data professional, **locate** data, and it ends when you **present** data analysis to stakeholders. The journey consists of five key stages: **collect, process, store, analyze, and activate**.

In this course, we'll focus on the first three stages—collect, process, and store—while the last two, **analyze and activate**, will be covered in a separate course.



### 1. Collect

The first stage is **collecting data**. Here, you gather the data you need for your analysis. This step involves identifying and sourcing relevant data from various

systems or datasets.

## 2. Process

The second stage is **processing** or transforming the data. In this stage, you review and explore the data to identify issues. You then **clean**, **organize**, and **standardize** it to make it ready for analysis. This step ensures the data is in a usable format for the next stages.

## 3. Store

Once the data is processed, you move to the **storing** stage. In this stage, you decide where to keep the data based on business needs. The data can be stored either **locally** or in the **cloud**, and the storage solution may vary based on security, accessibility, and volume requirements.

## 4. Analyze

In the **analyze** stage, you examine the data to identify trends, patterns, and insights. The goal here is to uncover the information that is critical to making informed decisions.

## 5. Activate

The final stage is to **activate** the data, where you present your findings and visualizations to stakeholders. This is the decision-making phase where actions are taken based on the insights provided.

The data journey is not always a linear process. It's **iterative**, meaning you might need to revisit earlier stages. For instance, after processing data, you may realize you need to collect additional data. Each project is unique, so the process might vary in complexity and time spent on each stage.

By understanding and mastering these stages, you create a solid foundation for **analyzing data** and delivering valuable insights to your organization.

## ▼ Steps for Effective Data Collection

As a cloud data professional, the first step in creating a dashboard or any data-driven project is **data collection**. This process is essential because it provides the foundation for your analysis and visualizations. Here are the **four key steps** to follow for effective data collection:

### 1. Identify Specific Questions

Before you start gathering data, you need to know what you're looking for. Begin by working closely with your **stakeholders** to identify the specific **business**

**questions** you need to answer. Understanding their needs will help guide your entire data collection process.

## 2. Data Discovery

Once the questions are clear, move into the **data discovery** phase. In this step, you'll explore and find the right data that can provide answers to those questions. Since data is often stored in different formats and locations, you'll also need to ensure that the data sources are aligned with the **metrics** and **insights** you're aiming to measure.

## 3. Data Gathering

The third step involves **gathering** the data from multiple sources. Data can exist in various formats (spreadsheets, databases, APIs, etc.), so you'll need to evaluate how to work with each data source. Consider how frequently the data needs to be updated and the effort required to maintain it. After locating the data, collect it into a single **staging area** where it can be organized for the next steps.

## 4. Data Staging

Finally, you'll need to **stage** the data. This involves bringing all the gathered data into a unified, usable staging area where it can be cleaned, transformed, and organized. Once your data is properly staged, it's ready to move into the next phase of the data journey: **processing**.

By following these steps, you can establish a strong foundation for **accurate analysis** and **effective visualizations**. Effective data collection is crucial for the success of any data-driven project, so becoming comfortable with this process is key.

### ▼ The Process Stage: Ensuring Clean and Consistent Data

Now that you've collected the necessary data for your sales dashboard, the next step is **processing** that data to make it usable. Raw data often comes with issues such as **incomplete records, duplicates, or incorrect data**. If left unchecked, these problems can lead to inaccurate analyses and faulty visualizations. Therefore, the process stage is crucial to ensure the data is **clean, consistent, and error-free**.

## What Is Data Transformation?

**Data transformation** is the process of converting raw, unorganized data into a format suitable for analysis. This includes changing inconsistent data formats into one cohesive structure and ensuring the data is error-free. It sets the foundation for successful data analysis and visualization development.

## Data Transformation vs. Data Processing

While the terms "data transformation" and "data processing" are sometimes used interchangeably, they aren't the same. **Data processing** is a broader term that covers various activities like data collection, cleaning, transformation, analysis, and visualization. **Data transformation**, on the other hand, refers specifically to the act of converting data from one format or structure to another to make it usable.

## Types of Data Transformation

There are six common types of data transformation techniques:

1. **Data Smoothing**: Reducing noise or variations in the data.
2. **Attribution Construction**: Creating new attributes from existing data.
3. **Data Generalization**: Replacing detailed data with a more abstract or general version.
4. **Data Aggregation**: Summarizing data by grouping it into a more meaningful structure.
5. **Data Discretization**: Converting continuous data into discrete intervals.
6. **Data Normalization**: Adjusting data to a standard scale or format.

The exact method you choose will depend on the **type of data** you're working with and how it will be used for analysis.

## Manual vs. Automated Data Transformation

Data transformation can be done in two main ways:

1. **Manual Transformation**: Using SQL or another common programming language to manually transform the data.
2. **Automated Pipelines**: Setting up automated data pipelines that streamline the transformation process.

The method you choose depends on the project's needs and the complexity of the data. Whether you opt for manual or automated transformation, it's crucial to build a **strong, clean foundation** that will set up your data analysis and visualizations for success.

Taking the time to ensure your data is thoroughly processed in this stage will pay off, helping you avoid errors and achieve reliable insights for stakeholders.

### ▼ Manual vs. Automated Data Transformation

As a cloud data professional working on a project, such as creating a sales dashboard, you'll need to decide between **manual** and **automated** data transformation methods. Both aim to prepare the data for analysis, but they differ in how they're implemented.

## Manual Data Transformation

Manual transformation involves using **programming languages** without relying on specialized software tools. The most common languages include:

- **SQL (Structured Query Language)**: Used to query and manage data in relational databases.
- **Python**: A high-level, versatile programming language, often used with libraries like **Pandas** for data manipulation and visualization.
- **R**: A programming language focused on statistical computing and data analysis, offering functions like `arrange`, `filter`, and `select` to transform datasets.

While manual transformation can handle both small and large datasets, it is typically better suited for **smaller datasets** due to the effort required. Writing code manually can be time-consuming, and it involves **testing, troubleshooting, and maintaining** the code to ensure accuracy.

## Automated Data Transformation

Automated transformation uses **processing and scripting tools** that simplify or eliminate the need for extensive programming. These tools can be integrated into workflows, significantly **reducing the effort** compared to manual coding.

While automated tools require less coding, you may still need to use languages like **SQL** or **Python** to make modifications within the tool. Automated transformation tools are especially beneficial for **large datasets** or high-velocity data because they are faster and more accurate than manual methods.

Automated tools can operate **locally** or in the **cloud**, depending on the organization's infrastructure. However, access to these tools may depend on their **availability and cost**, as they often require investment.

## Factors Influencing Your Choice

The choice between manual and automated transformation depends on several factors:

1. **Dataset size**: Large datasets are better handled by automated tools, while manual transformation works well for smaller datasets.
2. **Processing time**: If speed is a priority, automated methods are preferable for quick and accurate data transformation.
3. **Tool availability**: If automated tools aren't available due to budget constraints, manual transformation may be your only option.

4. **Combination of methods:** Even with automated tools, you might need to apply manual transformations for specific adjustments.

## Conclusion

Both methods have their strengths, and over the course of your career as a cloud data professional, you'll likely use **both manual and automated approaches** depending on the project's requirements.

# ▼ Handle raw data with data pipelines

## ▼ Data Pipelines in the Cloud

In cloud data analysis, a data pipeline works much like an assembly line in a factory, where raw materials (data) are transformed and transported to the final destination for storage and analysis. The goal of a data pipeline is to move data from different sources, ensuring it is processed **efficiently** and **consistently**.

There are **three main stages** in a data pipeline:

1. **Extract**
2. **Transform**
3. **Load**

Let's explore these stages in more detail:

### 1. Extract

The **extract** stage involves retrieving data from various sources and moving it to a temporary staging area. This data can come from multiple formats, such as **CSV files, databases**, or even cloud storage. In this stage, the data team identifies the raw data required for the project.

**Example:** An animal rescue organization creates a data pipeline to manage pet information. The data team pulls data from two sources: a CSV file with pet microchip registration numbers and a database with pet owner contact details. The raw data is moved to a staging area for the next stage.

### 2. Transform

Once extracted, the raw data enters the **transform** stage, where it's cleaned and put into a consistent format. During this stage, any **errors, duplicates, or inconsistencies** are addressed to ensure the data is usable for analysis.

**Key tasks** during this stage include:

- Cleaning up data to remove duplicates.
- Standardizing data formats.

- Enriching or modifying the data to suit specific requirements.

### 3. Load

After transformation, the data is ready to be loaded into its **final destination**, such as a database, data warehouse, or data lake. In the **load** stage, the processed data is inserted into a structured storage system where it can be used for analysis, visualization, and reporting.

**Example:** In the animal rescue organization, after transforming the microchip and pet owner data, it is loaded into a **data warehouse**. This makes it easier for the team to analyze the data, generate reports, and plan future actions.

## Flexible Pipelines

Just like an assembly line, data pipelines are designed to be **customized** to meet the specific needs of each project. While most pipelines follow the **extract, transform, load (ETL)** model, the **order of operations** or specific steps within each stage may vary depending on the **type of data** and the **goals of the project**.

### Benefits of Data Pipelines:

- **Automation:** Pipelines automate the movement of data, saving time and resources.
- **Improved Accuracy:** By standardizing data in the transform stage, errors are reduced.
- **Scalability:** Pipelines make it easier to handle **large amounts of data**, ensuring consistent and reliable results.

Data pipelines are critical tools for cloud data professionals, ensuring data flows smoothly and is always available for analysis and visualization in a clean and efficient manner.

## ▼ The Difference Between ETL and ELT

When designing a data pipeline, choosing between **ETL** (Extract, Transform, Load) and **ELT** (Extract, Load, Transform) is crucial for optimizing data flow and meeting specific business needs. Both pipelines aim to move data efficiently, but the order in which they handle **transformation** and **loading** stages differs, impacting performance, scalability, and flexibility.

### ETL (Extract, Transform, Load)

ETL has been a traditional method for **data integration** for decades and involves the following stages:

1. **Extract:** Data is collected from multiple sources, such as databases, CSV files, or APIs.

2. **Transform:** Before loading the data, it is cleaned, standardized, and put into a format that is suitable for analysis.
3. **Load:** After transformation, the data is inserted into the destination, such as a data warehouse or database.

#### When to use ETL:

- ETL is best suited for **on-premises** environments or when using structured data.
- It ensures that **transformed, clean data** is loaded into the storage system, reducing the complexity of further analysis.
- ETL is ideal when data transformations need to happen **before analysis**.

**Example:** A university may use ETL to integrate data from its student records, financial systems, and course catalogs, ensuring the data is transformed into a unified format before being loaded into a warehouse for tracking enrollment and financial trends.

## ELT (Extract, Load, Transform)

In ELT, the **loading** and **transformation** stages are swapped:

1. **Extract:** Like ETL, data is extracted from various sources.
2. **Load:** The raw data is **immediately loaded** into a data lake or data warehouse before any transformation occurs.
3. **Transform:** Once the data is stored, transformations happen on demand, depending on the analysis or business needs.

#### When to use ELT:

- ELT is typically used in **cloud-based environments** where storage and compute resources can easily scale to handle large datasets.
- It is preferred when working with **big data** and real-time or near-real-time processing because it allows data to be **analyzed quickly** before transformation.
- ELT gives flexibility to transform data **later**, which is helpful if the transformation needs are not clear at the outset.

**Example:** A manufacturing company tracking real-time data from sensors and production systems may opt for ELT, allowing them to load the data immediately and analyze it in near real-time. Transformation can be deferred to a later stage, giving them the flexibility to adapt to changes in the analysis needs.

## Key Differences:

- **Order of Transformation:** In ETL, transformation happens **before loading**, while in ELT, it occurs **after loading**.
- **Performance and Speed:** ELT is better for **large datasets or real-time data** where quick analysis is required. ETL is slower because transformation occurs before data is loaded.
- **Scalability:** ELT takes advantage of the **scalability of cloud platforms**, making it ideal for large, unstructured datasets.
- **Flexibility:** ELT offers more flexibility, as you can choose when and how to transform data after it is loaded.

Both ETL and ELT are critical tools in a data analyst's toolbox, and the choice between them depends on the specific **data needs, business goals, and available infrastructure**.

## ▼ Data Ingestion Methods That Suit Your Needs

Organizations rely on a multitude of data sources to make informed decisions. These sources can include websites, point-of-sale systems, social media platforms, and machinery sensor data, among others. Each source contributes to the overall data landscape, but to gain comprehensive insights, organizations must effectively consolidate these data sources. This is where **data ingestion** comes into play.

**Data ingestion** is the process of collecting data from various sources and moving it to a staging area, which is a crucial first step in preparing data for further processing and analysis. However, not all data ingestion techniques are created equal, and the choice of method should align with specific data requirements, particularly regarding the **time sensitivity** of the data.

## Understanding Time Sensitivity

Time-sensitive data is information that needs to be acted upon quickly; otherwise, its value diminishes. For instance, when planning a time-sensitive travel itinerary, you must consider how quickly you need to arrive at your destination. Different modes of transportation illustrate this concept well:

- **Buses:** These are cost-effective and suitable for non-urgent situations where you can afford to wait.
- **Taxis:** For urgent travel, taxis offer immediate transport without delays.

In data ingestion, the analogy holds:

- **Batch Ingestion:** Ideal when data processing can wait a bit.
- **Streaming Ingestion:** Best for immediate action on time-sensitive data.

## Data Ingestion Techniques

### 1. Batch Ingestion

Batch ingestion is a method where data is collected over time and processed in groups, known as **batches**. Here's how it works:

- Data is collected and stored temporarily.
- Processing occurs on a regular schedule (e.g., hourly, daily).

#### Advantages:

- **Economical:** Requires less computing power and storage, making it cost-effective for high-volume data.
- **Efficient for Non-Critical Data:** Well-suited for situations where immediate processing isn't necessary.

#### Example:

A large international non-profit organization collects donation data daily from its website and over the phone. With an average of over 1,000 donations received each day, the organization uses batch ingestion to collect the previous 24 hours' donation data at a scheduled time. This batch is sorted, grouped, and loaded into a staging area for further processing. The resulting insights help the organization analyze donation trends, such as the number of daily donations and average donation amounts.

### 2. Streaming Ingestion

Streaming ingestion involves collecting and processing data as soon as it becomes available. This method is best for organizations that require timely processing and quick decision-making.

#### Advantages:

- **Real-Time Data Processing:** Enables organizations to act on data immediately.
- **Proactive Monitoring:** Ideal for situations requiring near-real-time analysis.

#### Example:

A pharmaceutical manufacturing company monitors dryer temperatures with sensors. By employing streaming ingestion for this temperature data, the company can track temperature changes in real-time. If the temperature deviates from safe thresholds, the company can swiftly take corrective action to maintain product safety and quality.

## Choosing the Right Ingestion Method

Understanding the characteristics of batch and streaming ingestion is essential for making informed decisions about data ingestion:

- **Batch Ingestion** is ideal for scenarios where immediate processing is not critical, making it suitable for large datasets that can be aggregated over time.
- **Streaming Ingestion** is perfect for situations where data must be processed and acted upon without delay, such as real-time monitoring and immediate analytics.

By assessing your data needs and the time sensitivity of your data, you can choose the most appropriate ingestion method to ensure you receive the data you need when you need it.

## ▼ Data Mapping Helps Ensure Consistent Data

Data mapping plays a crucial role in the data pipeline, facilitating the integration and analysis of diverse data sources. Though it can be a complex process, you may already be familiar with the concept through everyday experiences. For example, think about birdwatching. When you encounter a new bird species, you identify it by observing its size, shape, color, markings, behavior, and call. You then compare these observations to a field guide to find a match. This process of comparison and identification mirrors the concept of data mapping.

### What is Data Mapping?

Data mapping is the process of matching fields from one data source to another, ensuring that information is aligned and standardized. Similar to identifying birds with a field guide, data mapping involves comparing data attributes to a known framework or schema. For data analysts, mapping data after ingestion is essential for easy understanding and analysis, ultimately leading to consistent and standardized data.

## How Data Mapping Works

To illustrate the process of data mapping, let's consider an example involving a public library's data team that seeks to combine data from two sources to track a twelve-month history of book circulation and new patron requests.

### 1. Identify the Data Sources:

- **Library Catalog:** Contains fields such as International Standard Book Number (ISBN), title, author, publisher, and publication date.
- **Circulation Database:** Includes barcode, title, author, and due date for each book.

### 2. Mapping the Fields:

- The data team identifies the fields that need mapping, including ISBN, title, author, publisher, and publication date.

### **3. Standardize Naming Conventions:**

- The title field in the circulation database is labeled "Book Title," while in the library catalog, it is simply "Title." The team decides to standardize this to "Title" for consistency.

### **4. Create Data Mapping Rules:**

- The team establishes rules for how fields will match. For instance, they need to convert the barcode in the circulation database to an ISBN number for accurate matching during analysis.

### **5. Testing the Mapping Rules:**

- Before applying the mapping rules to the entire dataset, the data team tests the rules on a small subset of data to ensure they work correctly.

### **6. Defining the Data Map:**

- A comprehensive map is created that outlines how the fields from both sources relate to each other.

### **7. Combining the Data:**

- Finally, the data from both sources is combined into a single dataset, making it ready for processing and analysis.

## **The Complexity of Data Mapping**

While this example illustrates a relatively straightforward data mapping scenario, real-world applications can be significantly more complex. Data mapping may involve multiple sources, varying data formats, and intricate relationships between fields. Additionally, manual data mapping can be time-consuming and prone to errors, which is why many data teams opt for automated data mapping tools.

## **Choosing Between Manual and Automated Mapping**

The decision to use manual or automated data mapping depends on several factors:

- Data Structure:** The complexity of the data being mapped.
- Project Size:** Larger projects may benefit from automation to handle the volume.
- Available Tools:** The tools and technologies at the team's disposal can influence the approach.

## The Importance of Data Mapping

Regardless of the method used, data mapping is a fundamental aspect of the data pipeline. It ensures that data is standardized and consistent, facilitating easier use and analysis. Improved data quality through effective mapping can lead to better decision-making for organizations, enabling them to leverage their data for strategic advantage.

### ▼ Introduction to Profiling and Cleaning Data

When you visit a store, you'll notice shelves stocked with items of various shapes and sizes. But how does store management keep track of all these products? The answer lies in the process of conducting regular inventories. This involves counting items, ensuring everything is organized, and confirming that the expected quantity of products is available on the shelves or in the warehouse. Similarly, **profiling and cleaning data** can be likened to taking an inventory of a store's items.

### What is Data Profiling?

Data profiling is the process of examining data to identify quality issues. It involves gathering information about the data's structure, format, values, and relationships. During this process, several quality issues can arise, including:

- **Missing Values:** Instances where data is absent or not recorded.
- **Duplicate Records:** Repeated entries that can skew analysis.
- **Inaccurate Data:** Incorrect or misleading information.
- **Inconsistent Data Formats:** Variations in how data is presented, making it difficult to analyze.

### What is Data Cleaning?

Once data profiling has identified these quality issues, the next step is **data cleaning**. This process involves fixing or removing data quality problems to ensure that the data is accurate, consistent, and complete.

### A Practical Example: Arpa the Data Analyst

To illustrate how data profiling and cleaning work in practice, consider Arpa, a data analyst for a retail chain. Each month, stores in the chain send Arpa data about their stock. Arpa's responsibilities include preparing this data for storage and analysis, which involves profiling and cleaning it.

#### 1. Data Profiling:

- Arpa starts by profiling the data to verify its accuracy and completeness before transformation, storage, and analysis.

- While data profiling can be done manually, using a profiling tool is typically more efficient.
- For instance, Arpa might use a tool to identify various columns in the dataset, such as item name, description, price, and quantity in stock.
- She can also determine the data type for each column, like using a string for the item name and a number for stock level. Additionally, Arpa checks for missing or duplicate values.

## 2. Data Cleaning:

- After profiling, Arpa can begin the cleaning process, addressing any quality issues discovered.
- For example, she may find duplicate item names or missing prices.
- Arpa's goal is to fix these issues to ensure the dataset is complete and ready for analysis.

## The Importance of Data Profiling and Cleaning

In summary, data profiling and cleaning are crucial steps in preparing data to enhance its quality. By routinely profiling and cleaning data, cloud data analysts can ensure that organizations have the best possible data to make informed decisions. This practice not only improves data accuracy and reliability but also supports better business outcomes. Regular attention to data quality ultimately empowers businesses to leverage their data effectively for strategic decision-making.

## ▼ Common Ways to Manipulate Data

When envisioning a data pipeline, one might picture a linear process where data is collected, processed, and analyzed sequentially. However, data pipelines are often more intricate and adaptable. The sequence of processes and techniques employed by a data analyst varies based on the unique requirements of each project.

## Example Scenario

Imagine you're a data analyst for a large school district, tasked with managing a dataset of student information that contains errors and inconsistent formats.

Typically, it's advisable to clean the data before manipulation. However, when the dataset is rife with errors, it might be more efficient to first manipulate the data to remove the most glaring issues. This approach can expedite the cleaning process and enhance the accuracy of the results.

As a cloud data analyst, it's crucial to understand the various techniques available and how to apply them effectively to achieve the desired data outcomes. Here,

we'll explore three common techniques for manipulating data: **data standardization**, **data enrichment**, and **data conversion**.

## 1. Data Standardization

Data standardization ensures that all data within a dataset adheres to a common format. This consistency enhances reliability, making the data easier to process and analyze.

### Example:

Kyle, a data analyst managing the product catalog for a large online retailer, observes that product names are formatted inconsistently—some are in uppercase, while others are in lowercase. To standardize the data, Kyle employs a data standardization tool to convert all product names to lowercase. This step enhances the consistency of the data, facilitating smoother processing and analysis.

## 2. Data Enrichment

Data enrichment involves adding supplementary information to existing data, either by creating new fields or by merging with other data sources.

### Example:

After standardizing the product names, Kyle seeks to add Stock Keeping Unit (SKU) numbers to each product to better track inventory. He utilizes a data enrichment tool to join product names in the database with their corresponding SKUs from a product catalog. This integration provides each product with an SKU, aiding the retailer in effectively managing inventory and sales.

## 3. Data Conversion

Data conversion is the process of changing the format of data to enhance compatibility, readability, or security. This may involve transforming data for use in different systems, optimizing storage space, or making it easier to interpret.

### Example:

Given the large and complex nature of the data, Kyle recognizes the need to compress it before transferring it to the storage destination. He uses tools to read the CSV files and convert them into the Parquet format, an open-source file format that stores data in a columnar structure. This conversion optimizes the data for efficiency and analysis.

## Conclusion

In summary, **data standardization**, **data enrichment**, and **data conversion** are essential techniques for manipulating data, making it more functional and accessible. As a cloud data analyst, the choice of techniques will ultimately depend on the specific characteristics of the data and the business objectives at hand. By

understanding the various options and their applications, analysts can prepare and manipulate data effectively, ensuring it is in the best possible condition for storage and analysis.

## ▼ Different Approaches to Loading Data

In a data pipeline, data typically progresses through three key stages: **Extract**, **Transform**, and **Load (ETL)**. Each stage plays a crucial role in preparing data for analysis:

- **Extract:** Data is collected from various sources.
- **Transform:** Data is cleaned, manipulated, and enriched.
- **Load:** Data is moved to a new location where it can be utilized for analysis.

To illustrate the loading stage, consider an example involving donating items to a local charity:

1. **Collecting Items:** This represents the extract stage.
2. **Cleaning Items:** This is akin to the transform stage.
3. **Loading Items onto the Truck:** This mirrors the load stage.

## Understanding the Load Stage

The load stage involves transferring data into its destination storage. Although this stage typically concludes the data pipeline, it can sometimes occur earlier, as seen in **ELT (Extract, Load, Transform)** pipelines, where data is loaded before it is transformed. Regardless of its timing, the load stage aims to transfer data from a staging area to permanent storage.

## Preparing for Data Loading

Before loading data into storage, it's essential to prepare the destination. This preparation ensures that the storage system can effectively receive the incoming data. Depending on the dataset, this preparation may involve:

- Creating new tables or directories.
- Configuring the storage to accept the incoming data.

Once the destination is ready, you can proceed to load the data.

## Common Loading Methods

There are three prevalent methods for loading data:

### 1. Batch Loading

- **Description:** Data is transferred in groups called batches according to a predetermined schedule.

- **Pros:**
  - Efficient for handling large datasets.
- **Cons:**
  - High data volume can overload the destination storage.

## 2. Streaming Loading

- **Description:** Data is continuously moved to storage as it becomes available, enabling near-real-time access.
- **Pros:**
  - Ideal for time-sensitive data, ensuring timely decision-making.
  - Reduces the risk of overloading the destination system by processing data continuously.
- **Cons:**
  - Requires systems capable of handling continuous data flow.

## 3. Incremental Loading

- **Description:** Only data that has changed since the last load is transferred to storage.
- **Pros:**
  - Saves time and resources, particularly beneficial for large datasets that frequently change.
- **Cons:**
  - Frequency of incremental loads depends on dataset size, change frequency, and performance requirements of the destination system.

## Final Data Integrity Check

After the loading process, it's critical to verify the integrity and accuracy of the data. This final check ensures the data is ready for analysis and helps maintain high data quality.

## The Complexity of Data Loading

Loading data can be a complex process, especially when dealing with large datasets. Many data teams utilize automated tools to streamline loading and minimize the risk of data loss or errors. Even with these tools, understanding the data loading process is essential, as it directly impacts the quality of data available for analysis and influences decision-making across the organization.

## ▼ Overview of Data Validation Strategies and Rules

An essential aspect of any assembly line is the **quality control inspector**, who ensures that all products meet the required standards before they reach consumers. For instance, a quality control inspector on a toy car assembly line checks each car for defects like missing or damaged parts and compares them to specifications to confirm they meet standards. This inspection process is akin to **data validation** in a data pipeline.

## What is Data Validation?

**Data validation** is the process of checking and rechecking the quality of data to ensure it is complete, accurate, secure, and consistent. It is a critical part of the **Extract, Transform, and Load (ETL)** stages of a data pipeline. Although data validation can occur at any stage, it is especially crucial during the load stage, as it represents the last opportunity to correct errors before the data is used for analysis and reporting.

## Common Data Validation Techniques

The specific techniques a data analyst uses to validate data will depend on the data being processed and the organization's business needs. Here are some common validation methods:

### 1. Type Validation

- Ensures that the data is of the correct type.
- **Example:** In a donor database, verifying that the donor zip code field is a number data type rather than a string.

### 2. Format Validation

- Confirms that the data follows a specific format.
- **Example:** Checking that date values are consistently formatted across a field.

### 3. Uniqueness Validation (Duplicate Validation)

- Checks for duplicate records to ensure that each entry is unique.
- **Example:** Ensuring no two donors have the same name or email address in the donor records.

### 4. Range Validation

- Verifies that data falls within a specified range.
- **Example:** Ensuring that the age of a donor is between 0 and 130.

### 5. Null Validation

- Identifies empty or missing values, which can create issues in analysis and reporting.
- **Example:** Checking for null values in critical fields, such as donor names or donation amounts.

## **Handling Data That Cannot Be Validated**

When data cannot be validated, the approach taken depends on the specific validation rule in place:

- **Data Discarding:** In cases where errors are too complex or if the data is non-essential, it may be discarded. For instance, a donor's age outside the valid range might be removed from the dataset.
- **Flagging Data:** Invalid data may be loaded into storage with a flag. This alerts users that the data requires attention before it can be used. For example, a misspelled donor's email address might be flagged for manual correction.
- **Automatic Correction:** Some systems allow for automatic correction of data errors. For instance, if a donor's zip code is invalid, it could be matched against a valid address database to correct the mistake automatically.

## **Conclusion**

In summary, data validation is akin to quality control for data. As data moves through a pipeline, ensuring its completeness, accuracy, and consistency is crucial. By employing various validation techniques, data analysts can guarantee that the data used for analysis is reliable, ultimately leading to more accurate insights and better decision-making based on that data.

## **▼ Cloud data optimization strategies**

### **▼ Challenges of Data Transformation**

In this discussion, we will explore the challenges that data teams face when transforming data in the workplace and emphasize the importance of having a reliable and high-quality data transformation plan.

## **The Importance of Data Transformation**

Data is a crucial raw material that informs business decision-making. However, the volume of data that organizations manage and store has increased significantly in recent years. As a cloud data analyst working with large datasets, uncovering the value of this data is essential for providing critical insights to your data team and stakeholders. Despite its importance, implementing a data transformation plan poses several challenges, particularly concerning resources and data integrity.

## **Challenges Related to Resources**

## **1. Computational Power**

- Data transformation can be resource-intensive, requiring significant computational power to process large datasets.
- Historically, this necessitated investment in expensive computer hardware capable of handling heavy computational loads.
- However, with the rise of cloud computing, organizations can now access virtual computational power, reducing the need for costly hardware investments. Despite this advantage, it's crucial to factor in the costs of cloud services when planning your project.

## **2. Storage Requirements**

- Data transformation processes typically use ETL (Extract, Transform, Load) or ELT (Extract, Load, Transform) pipelines, which facilitate the movement and processing of data.
- Often, not all stored data is used, leading to unnecessary storage fees.
- To mitigate these costs, organizations must actively manage their storage, including regularly reviewing and deleting unneeded data.

## **3. Time and Skilled Personnel**

- Transforming data is a time-consuming process that requires a team of skilled professionals to execute the transformation plan effectively.
- For organizations with smaller budgets, acquiring the necessary talent and resources can be a significant challenge.
- Utilizing cost-effective cloud services can help alleviate some of these resource-related issues.

## **4. Variety of Tools**

- Different tools are available for data transformation, and their effectiveness varies depending on the task.
- As a cloud data analyst, understanding the available tools and selecting the right ones for specific needs is essential for successful data transformation.

## **Challenges Related to Data Integrity**

While resource management is a significant concern, **data integrity** presents one of the most critical ongoing challenges for data teams. Even minor errors in data can profoundly impact the outcomes of a data transformation project.

### **1. Definition of Data Integrity**

- Data integrity refers to the accuracy, completeness, consistency, and trustworthiness of data throughout its life cycle. Errors can arise before and during the transformation process.

## 2. Sources of Errors

- **Data Entry Mistakes:** Typos and other errors during data entry can introduce significant issues.
- **Bulk Reading Errors:** Automated processes may introduce errors if the data is not adequately cleaned before transformation.
- **Transformation Errors:** Errors can also occur during the transformation phase itself, such as:
  - Incorrect data type conversions
  - Missing values
  - Incorrect formatting
  - Misleading results due to inappropriate aggregation methods

## 3. Consequences of Errors

- If these errors are not identified before the data is stored and analyzed, they can compromise the integrity of the data used for reporting and decision-making.

## Strategies for Managing Data Integrity

While it is impossible to eliminate errors entirely, having an effective plan in place to manage data integrity is crucial. Such a plan should focus on minimizing mistakes and ensuring stakeholders have access to the highest quality data to inform their decisions.

## Conclusion

As a cloud data analyst, the responsibility of finding value in the increasingly complex and vast amounts of data that flow into an organization can be daunting. A robust data transformation plan is essential for managing resources and maintaining data integrity. This will ultimately empower you to provide critical insights that support informed decision-making and drive business growth.

## ▼ Data Aggregation: Unlocking Meaningful Insights

In this discussion, we'll explore how data aggregation can facilitate the extraction of meaningful insights from vast amounts of data.

## The Importance of Data Aggregation

Data is continuously generated from various sources such as smart devices, machines, websites, social media, and videos, often arriving in near-real time. This data presents an opportunity to cultivate a data-driven culture within organizations, but it also poses significant challenges for data analysts due to its sheer volume, variety, and velocity.

Data aggregation emerges as a powerful technique to manage this complexity. It involves gathering data and expressing it in a summarized form, allowing organizations to derive valuable insights in three key ways:

- 1. Managing Data Effectively**
- 2. Making Data More Accessible**
- 3. Observing Data Trends**

## **1. Managing Data Effectively**

The increasing volume and velocity of data have led businesses to acquire more storage solutions. However, not all data stored is relevant or necessary for long-term retention.

- **Pro Tip:** Irrelevant data occupies valuable storage space and complicates the process of extracting insights.
- **Solution:** Aggregation helps reduce data clutter by removing irrelevant data and retaining only meaningful information.

**Example:** When a user clicks a link on a website, numerous data points are generated (e.g., time of click, IP address, browser type). Instead of storing all this raw data, businesses can focus on key metrics, such as the number of clicks per hour. By aggregating this data, the organization can store a single data point—the average number of clicks—rather than the complete set of raw data. This approach streamlines data management and optimizes storage.

## **2. Making Data More Accessible**

Aggregated data simplifies the analytics process for data teams.

- **Efficiency:** Instead of sifting through extensive raw datasets, teams can query pre-aggregated metrics. This saves time and effort.
- **Example:** The average number of clicks per hour can be quickly accessed without the need for further calculations, enabling data teams to focus on analysis rather than data processing.

## **3. Observing Data Trends**

Data aggregation also facilitates the identification of trends, which is crucial for informed decision-making.

- **Trend Analysis:** With easily accessible aggregated data, data teams can chart metrics over time to spot patterns and trends.
- **Example:** By analyzing the average number of clicks per hour, the marketing team can determine peak times for website traffic, helping them strategize ad buys and forecast future traffic.

## Conclusion

Data aggregation is a versatile and powerful tool within an organization's data transformation plan. By effectively managing data, making it more accessible, and revealing trends, aggregation enhances the decision-making process and drives business success. The benefits of implementing data aggregation strategies are extensive, making it an essential component for any data-driven organization.

## ▼ Consequences of Duplicate Data and How to Eliminate It

In this discussion, we will examine the common issue of duplicate data that cloud data analysts frequently encounter, along with its consequences and strategies for elimination.

### Understanding Duplicate Data

Duplicate data refers to records that repeat information, either in whole or in part. There are two primary types of duplicates:

1. **Partial Duplicates:** Records where only a part of the data repeats.
2. **Exact Duplicates:** Records where all data is identical.

#### Example of Partial Duplicates:

Consider a science organization's database that includes information about its members. If there are two records for a member named "Eli Arnez" residing at the same Madrid address, but one record has the phone number "1223 555" while the other has no phone number, this represents a partial duplicate.

#### Example of Exact Duplicates:

If the same organization has one entry for a member with identical information (name, address, and phone number) entered twice, this is an exact duplicate.

### Consequences of Duplicate Data

Duplicate data can severely undermine data integrity, leading to various negative outcomes:

1. **Inaccurate Reporting:** When duplicates exist, the analysis may yield skewed results, leading to incorrect business decisions.

- **Example:** If an analyst calculates the average price of merchandise sold and finds an unexpectedly high value due to one shirt being recorded three times, the aggregated data becomes misleading. Once duplicates are removed, the accurate metric can be obtained.
2. **Wasted Resources:** Duplicate records can result in unnecessary expenditure, particularly in marketing and communications.
    - **Example:** If a marketing team plans to send advertisements to all members based on a dataset, duplicates can cause some members to receive multiple ads, wasting the marketing budget. Removing duplicates ensures each member receives only one advertisement.
  3. **Increased Storage Costs:** Duplicates consume storage space, leading to redundant data that inflates storage requirements and costs.

## Strategies to Eliminate Duplicate Data

To maintain data quality and prevent the aforementioned issues, it is essential to implement effective deduplication strategies.

### 1. Manual Deduplication:

- Analysts can manually compare rows of data to identify duplicates.
- **Pros:** This method is effective for smaller datasets.
- **Cons:** It can be time-consuming and inefficient for larger datasets.

### 2. Automated Deduplication Tools:

- Many data analysts use automated tools that employ algorithms to compare chunks or blocks of data for duplicates.
- **Pros:** These tools are much more efficient and can quickly identify duplicates, especially partial duplicates, in large datasets.

### 3. Deduplication as Part of Data Transformation Plans:

- Integrating deduplication into an organization's data transformation plan ensures ongoing data quality and helps maintain accurate insights.

## Conclusion

Duplicate data is a prevalent challenge that can lead to inaccurate insights, wasted resources, and increased costs. By implementing effective deduplication strategies, including both manual methods for small datasets and automated tools for larger datasets, data analysts can ensure data integrity and provide meaningful insights for decision-making. Addressing duplicate data is crucial for organizations aiming to leverage their data for growth and success.

## ▼ Overview: How Joins Combine Data from Different Tables

In this guide, we will explore the concept of joins, a powerful tool for data analysts used to combine data from different tables. Understanding how joins work and the challenges that come with them, such as missing data and null values, is essential for effective data analysis.

### Understanding Joins

Joining tables can be challenging due to two common issues:

1. **Missing Data:** Rows that are expected to be returned do not appear in the joined table.
2. **Null Values:** Indicate the absence of a value for a field, which can occur due to unmatched rows in the join.

### Types of Joins

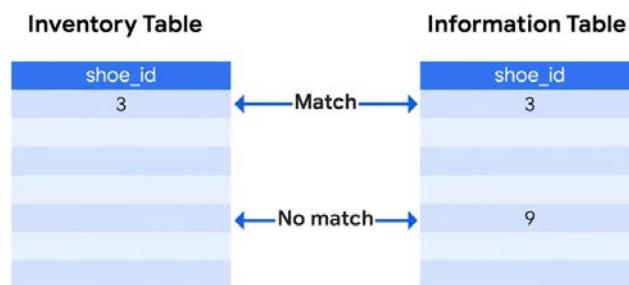
The type of join selected by a data analyst determines how missing data is handled. There are two primary types of joins to understand: **inner joins** and **outer joins**.

#### Inner Join

- An **inner join** returns only the rows that match in both tables.
- **Example:** Kenji, a data analyst, wants to join an inventory table and an information table using a common column, "shoe ID."

#### Scenario:

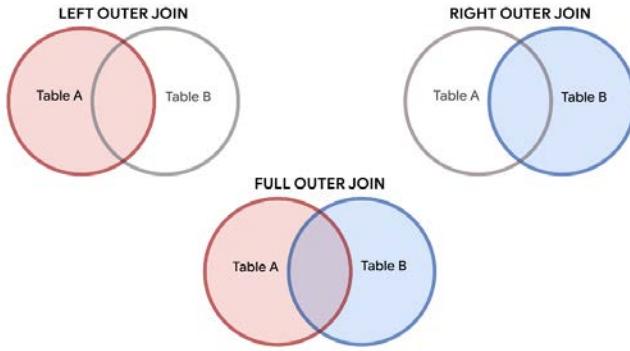
- If "shoe\_id" 3 exists in both tables, it will appear in the joined result. However, if "shoe\_id" 9 is only in the information table, it will be omitted from the results.



#### Outer Join

- An **outer join** returns both matched and unmatched rows from one or both tables.

- There are three types of outer joins:



- **Left Outer Join:** Returns all rows from the left table and matched rows from the right table.
- **Right Outer Join:** Returns all rows from the right table and matched rows from the left table.
- **Full Outer Join:** Returns all rows from both tables.

#### **Scenario:**

- Kenji decides to use a **left outer join** to include all rows from the inventory table and only matched rows from the information table. This results in more rows being returned, including those with unmatched data.

## **Handling Null Values**

In a left outer join, unmatched values in the left table are represented as null values in the joined table.

- **Definition of Null Values:** A null value indicates that there is no available value for a field, meaning no match was found.
- **Important Note:** Null values are not the same as zeros or blank data. This distinction can be useful for data analysts to indicate missing data points.

## **Challenges with Null Values**

While null values are expected behavior in outer joins, they can introduce complexity:

1. **Differentiating Nulls:** It may be difficult to distinguish between null values that result from no match in the join and those that existed in the original dataset.
2. **Impact on Analysis:** Analysts must consider how null values could affect their results and the insights derived from the data.

## Conclusion

Joins are a fundamental part of data analysis that enable the combination of data from different sources. By understanding the implications of different types of joins, as well as how to manage missing data and null values, data analysts can join tables more efficiently and generate meaningful insights. Ultimately, the best approach to joining data depends on the specific needs of the analysis and the characteristics of the dataset.

## ▼ Data Derivation: Combining Data to Obtain New Insights

To answer complex data questions, you often need to combine data in innovative ways. For instance, if you want to find the total points earned by the top three players in a chess tournament, you would need to identify their scores and sum them up. This process involves data derivation.

### What is Data Derivation?

Data derivation is the process of combining and processing existing data using algorithms to create new data. This new data can provide insights that are not directly available from the base data alone.

1. **Raw Data:** This is the base data that already exists within your dataset.
2. **Algorithm:** A set of rules or processes followed to derive new data from the raw data.

### The Data Derivation Process

The derivation process begins with raw data and utilizes an algorithm to transform this data into a new format or metric that can yield insights.

### Example: Anya's Shoe Company

- **Scenario:** Anya, the manager of a shoe company, needs a daily report to track shoes that have been on the shelves for 30 days or more to minimize profit loss from steep discounts.
- **Challenge:** The raw data needed to create this report doesn't currently exist in the dataset.
- **Data Exploration:** Kenji, the data analyst, discovers that warehouse workers record an "arrival date-timestamp" for each pair of shoes when they reach the shelf.
- **Creating Derived Data:** Kenji can use this timestamp data to derive how long each pair of shoes has been in the warehouse. He applies a simple algorithm to calculate the duration and formats this data into a report.

- **Output:** The derived data can then be queried like a regular database table, allowing Kenji to create a daily report for Anya. This report provides timely information on which shoes should be prioritized for sales to prevent value loss.

## Benefits of Derived Data

- **Enhanced Performance:** Working with derived data can increase analytical efficiency.
- **In-depth Insights:** Analysts can uncover insights that are not readily apparent from the base data alone.

## Challenges of Data Derivation

While derived data can provide valuable insights, there are some challenges to consider:

1. **Accuracy:** Derived data may not always be as accurate as the original base data. Errors in the algorithm or changes in the base data after derivation can affect results. Analysts should be cautious when creating new derived data from previously derived data.
2. **Data Ownership and Privacy:** When handling personally identifiable information (PII), it's essential to consider data ownership and privacy regulations. For instance, if a new warehouse worker provides PII during hiring, their consent applies only to the initial data collection. Any further use of this data must comply with relevant policies and regulations.

## Conclusion

Transforming existing data into new insights through data derivation is a powerful tool for cloud data analysts. It can help unlock insights that are not immediately available in the existing data, allowing analysts to tackle more complex questions. However, it's crucial to execute this process with care to ensure accuracy and compliance with data privacy regulations.

## ▼ 4- The Power of Storytelling: How to Visualize Data in the Cloud

### ▼ Introduction to the power of storytelling: How to visualize data in the cloud

#### ▼ Introduction: Data Visualization for Cloud Data Analysts

As a cloud data analyst, effectively communicating your findings to various stakeholders is a crucial aspect of your role. One powerful method of communication is **data visualization**. Visualizing data in the cloud not only helps your team better understand the data but also empowers stakeholders to make informed business decisions.

## The Power of Data Visualization

Data visualization involves the graphical representation of data using charts, graphs, and other visual formats. These tools allow users to interact with the data, uncover insights, and make decisions that are backed by solid analysis. This course will help you transform numbers and text into engaging visualizations that can inspire business leaders to act.

## Prerequisites

Before diving into this course, it's beneficial to have:

- An understanding of how data is stored and accessed on the cloud.
- Basic knowledge of **SQL** (Structured Query Language), which is used to query relational databases.
- Familiarity with querying basics to facilitate smoother learning.

## Course Overview

Let me introduce myself—I'm **C.J.**, a Cloud Customer Engineer at Google. I work with some of the largest retailers globally, helping them leverage Google Cloud's data services to gain valuable business insights. In this course, I will guide you through the essential tools and techniques for creating compelling data visualizations.

## What You'll Learn

- **Data Storytelling:** We'll begin by exploring how to craft narratives that captivate your audience using cloud-based insights.
- **UX/UI Design:** You'll learn the basics of user experience (UX) and user interface (UI) design to enhance your communication.
- **Visualization Planning:** We'll discuss how to translate data stories into effective visualization plans that meet specific business needs, focusing on how data types inform visual representation choices.
- **Working with Stakeholders:** You'll discover strategies for translating stakeholder requests into meaningful visualizations.

- **Hands-on Learning:** You'll practice accessing, exploring, and reporting on cloud datasets, working with dimensions, measures, and tools like **Looker** to build dashboards.
- **Large-Scale Data Visualization:** You'll explore tools designed to handle large datasets with integrated security features, fostering a data-driven culture within organizations.
- **Advanced Tools and Data Modeling:** You'll also step into the role of a developer, learning data modeling languages, writing dashboards as code, and addressing complex business needs.

## Why is This Important?

Learning to create visual data that is accessible to both technical and non-technical users is a key skill for any cloud data professional. Your ability to present clear, insightful visualizations will improve data literacy across your organization and drive better business outcomes.

## ▼ Ways Cloud Data Enhances Storytelling

A captivating story requires engaging characters and a compelling setting. While fictional storytelling relies on imagination, **data storytelling** requires robust and relevant data. The introduction of **cloud data** has revolutionized how data stories are crafted by providing more options and greater context than working with local data alone.

## The Evolution of Data Storytelling with Cloud Data

Before the emergence of cloud data and cloud analysis tools, data analysts were restricted to working with **local data**. This meant they had to manage, track, and analyze only the data that was immediately available to them. However, with the advent of **centralized cloud data warehouses**, it is now possible to access **organization-wide data** as well as external sources, creating a more comprehensive narrative.

One of the primary advantages of cloud data is the ability to integrate a wider variety of data sources, including:

- External and public datasets.
- Application data from platforms like Salesforce and Google Analytics.

This expanded access enriches analysis and enhances the storytelling process.

## Breaking Down Data Silos

In traditional setups, data was often stored in isolated **data silos**. For example, a clothing retailer may store sales and inventory data on-site, while marketing teams

may track campaign data separately in Google Sheets. These data silos limit access across the organization, making it difficult for teams to collaborate and paint a full picture. Silos also increase the risk of data inconsistencies, which can compromise data integrity and create gaps in the narrative.

With **cloud-based data analysis**, data from different sources can be combined into a cohesive whole, offering a more complete and accurate view of organizational trends. This allows teams to:

- Understand the full context.
- Collaborate more effectively.
- Craft insightful data stories that drive informed decisions.

## Enhancing Analysis with Cloud Tools

Consider the same clothing retailer. By leveraging cloud tools, the data team can connect the sales and inventory databases of different stores. The marketing team can now see what products are selling well across various locations. Likewise, the purchasing department can make informed decisions about which brands to continue stocking. This real-time access to organization-wide data enables all departments to collaborate, improving performance and decision-making.

As a cloud data analyst, you can take this even further by connecting to **external data sources**—such as public datasets and third-party tools—to provide additional context and depth to your analysis. For instance, connecting to global clothing trend datasets can inform decisions about product development and marketing strategies.

## Real-World Impact of Cloud Data

The flexibility of cloud data allows for more dynamic and impactful storytelling. Here are a few practical examples:

- **Language Localization:** By analyzing regional data, a retailer can identify that certain promotions are underperforming because the marketing materials are not in the community's primary language. With this insight, the retailer can adjust their strategy to better reach those customers.
- **Advertising Effectiveness:** Cloud tools can also help monitor the performance of campaigns, such as tracking the success of **Google Ads** in specific regions. By combining this with historical sales data, a retailer can determine which locations are responding to their advertising efforts.

## Conclusion

Cloud data provides endless opportunities to enrich data storytelling. By integrating data from multiple sources and offering comprehensive insights, you can provide valuable context to different teams, such as sales and marketing, helping them make more informed decisions. As a cloud data professional, your ability to blend diverse data into a cohesive and compelling narrative will be invaluable to your organization.

## ▼ Planning Phases of the Data Visualization Workflow

As a cloud data analyst, it's crucial to build data visualizations based on thorough analysis and a well-defined narrative plan. Skipping this step is like embarking on a road trip without a map—without knowing your destination or what you'll need along the way. In this guide, we'll explore the essential planning phases of the data visualization workflow, focusing on how unique cloud features enhance digital consumption, user experience, and data governance.

### 1. Understanding Digital Consumption

The first step in the planning process is to grasp the **digital consumption** aspect. This means understanding the platform and medium through which users will engage with the data. You must consider how your audience will interact with the data and what they expect from their experience.

- **Know Your Audience:** It's essential to identify who will view your visualizations. Are they executives, analysts, or front-line staff? Each group will have different needs and expectations.
- **Expectations for Interaction:** Users in today's digital world anticipate up-to-date and interactive information. For instance, if you design a dashboard for a company with multiple locations, each store's team will likely focus on regional sales data, while central teams will need comparative insights across all locations.

### 2. Designing for User Experience

Creating an intuitive user experience is paramount. Users prefer tools that are easy to navigate and understand without extensive instruction. Here are some key points to consider:

- **Intuitive Design:** Visualizations should convey meaning and purpose quickly. Employing established design principles helps users become comfortable with the tool and engage with it effectively.
- **User-Friendly Features:** Implement interactive elements, such as filters and drill-down options, allowing users to tailor the information to their specific needs.

### 3. Data Governance and Security

Given the risks associated with accessing data over the internet, robust data governance and security measures are vital. Here's what to keep in mind:

- **Protection of Personally Identifiable Information (PII):** When designing visualizations that involve sensitive data, like sales staff contact information, ensure that only aggregated data is presented, safeguarding individual privacy.
- **Access Control:** Implement role-based access controls to determine who can view specific data. For instance, store supervisors may have access to detailed sales data, while team members might only see summary-level insights.
- **Strategic Data Handling:** Create a plan that outlines how data will be managed, shared, and protected throughout the visualization process.

## Conclusion

In summary, planning data visualizations in the cloud involves understanding digital consumption, designing an intuitive user experience, and prioritizing data governance and security. By applying these principles, you can ensure that your visualizations are effective, user-friendly, and secure. Just as a well-planned road trip leads to a successful journey, careful planning in data visualization will help you reach your analytical destination without any hitches.

## ▼ Dashboard Design for Effective Communication

As you embark on the journey of building dashboards, it's crucial to remember that the primary objective isn't the technology itself; it's all about **communication**. Let's delve into the key principles that can guide you in designing effective dashboards that tell a compelling data story.

### 1. Focus on Communication, Not Just Technology

While technical skills, tools, and coding are essential for creating dashboards, your main goal should be to convey the narrative that the data holds. The dashboard serves as a medium through which the data can communicate its story.

- **Data as a Story:** Think of each section of your dashboard, or each visualization, as a chapter in a book. Each component contributes to the overall narrative, building context and advancing the story. Ensure that every piece of data presented serves a purpose and is relevant to the narrative.

### 2. Clarity and Consistency

For effective communication, it's vital that all users interpret the data in the same way.

- **Shared Understanding:** Unlike literature, where interpretations may vary, data analysis requires a consistent understanding among users. Design your visuals and narratives so that everyone can derive the same insights from the data.

### 3. Visual Hierarchy

Pay attention to the size and placement of your visuals, as these elements direct user attention.

- **Emphasizing Key Insights:** Making one graph larger than others can highlight its importance. Use this technique strategically to guide users toward the most critical information on your dashboard.

### 4. Non-Technical Communication

Beyond technical data, dashboards also communicate various non-technical messages through design choices.

- **Tone and Language:** The words and tone you use can shape user perception. Complex terminology might suggest formality, while a casual tone can create a relaxed atmosphere.
- **Visual Style:** The graphics you choose should align with the tone you want to set—whether simple, sophisticated, or playful. This visual language contributes to how users feel about the data.

### 5. Brand Alignment

Ensure your dashboard reflects your organization's brand identity.

- **Cohesive Design:** Use official colors, logos, and design elements that resonate with your corporate culture. This alignment helps your dashboard feel like an integral part of the organizational toolkit.

### 6. Professionalism in Design

The quality of your design reflects your professionalism and the effort you've put into your work.

- **User Experience:** An effective dashboard not only presents data clearly but also shows your dedication to making insights accessible and actionable for users.

## Conclusion

Ultimately, the purpose of dashboards extends beyond the technology involved; it lies in effective communication. By focusing on how your designs can convey messages, maintain clarity, and align with your organization's brand, you'll be better prepared to create dashboards that resonate with users and facilitate

understanding. Harness these principles to enhance your skills as a communicative and effective data analyst, ensuring that your dashboards not only look professional but also serve their purpose as powerful storytelling tools.

## ▼ The Importance of UX/UI Design

In the professional landscape, individuals often specialize in various fields, each bringing unique talents to the table. For instance, teachers excel in education, musicians master instruments, and coders develop applications for user interaction. Imagine the complexity if everyone had to learn these skills! Fortunately, users can depend on data professionals to craft effective user interfaces that foster positive experiences. This video will explore the significance of User Experience (UX) and User Interface (UI) design.

### Definitions and Concepts

- **User Experience (UX):** This term refers to the overall experience a user has with a product or service, particularly regarding its ease of use and enjoyment. As technology evolves, UX continues to gain importance, making human-computer interactions more intuitive and user-friendly.
- **User Interface (UI):** UI encompasses all elements that facilitate user interaction with a computer system, including everything visible on a screen or device. It allows users to engage with software and hardware without requiring programming knowledge.

Both UX and UI design focus on the user's needs. As a data analyst, you'll frequently encounter these concepts together as UX/UI, both essential for your work with cloud visualizations.

### The Impact of UX/UI on Data Analysis

The effectiveness of your data visualizations relies heavily on the user experience. If stakeholders find your solution challenging to learn, cumbersome, or simply unattractive, they are unlikely to use it. The goal of data analysis is to provide actionable insights, especially through visualizations.

### UX/UI in Action

Consider your experiences with different websites or apps:

- **Negative Experience:** Recall a website where you struggled to find information, leading to frustration. The user experience was poor due to inadequate UI design.
- **Positive Experience:** Now think of a visually appealing app where the information you needed was easily accessible. This positive interaction is a result of effective UX/UI design.

## The Role of UX/UI in Data Visualization

Creating well-designed dashboards and reports is about more than aesthetics; it's about ensuring users can understand and utilize the data effectively. Keeping user needs at the forefront allows you to design impactful visualizations that bring data to life.

For example, if you want users to drill down into details within a dashboard, you must indicate that those sections are interactive. Visual cues, such as bold or underlined text and icons, can inform users about clickable options.

## Accessibility Considerations

While discussing visual design choices, it's essential to consider other aspects of design, including accessibility. Some users rely on screen readers or other assistive technologies. Therefore, it's crucial to accommodate diverse user needs in your design.

## Conclusion

UX/UI design plays a vital role in data analysis and visualization. By understanding and prioritizing user experience, you can create effective and professional dashboards that enhance data insights. Keep exploring UX and UI principles to continuously improve how you present data, ensuring that all users can easily navigate and comprehend the information you provide.

## ▼ Design Decisions for Data Visualizations

When developing data visualizations in the cloud, you'll encounter numerous critical design decisions. Each decision should focus on one essential question: **What's best for the audience?** Here are some key aspects to consider:

### Key Visualization Decisions

#### 1. Is it Insightful?

- The primary goal of any visualization is to convey meaningful insights to the audience.

#### 2. Does the Layout Make Sense?

- Logical organization is crucial. A well-designed visualization should be intuitive and easy for users to navigate.

#### 3. Is it Accessible for Everyone?

- Ensure that your visualizations cater to a diverse audience, considering various accessibility needs.

## Audience-Centric Design

As a cloud data analyst, your design choices should always benefit your audience. For instance, creating a visually appealing dashboard is not enough if it lacks logical organization. A cluttered or poorly arranged dashboard can hinder user engagement and comprehension.

## Organizing Dashboards Logically

To achieve logical organization, consider working backward. Start by envisioning what your audience needs and wants from the visualization. This approach helps inform your design decisions, aligning them with user goals and objectives.

## Research Phase: Understanding User Needs

During the research phase of data visualization planning, focus on understanding user requirements through usability studies or user research. Various methods can be employed to gather insights, depending on factors like the industry, budget, and available tools.

## Technical Aspects and Workflows

As you develop the technical aspects of your design, continue to prioritize your audience's needs. **Key considerations include:**

- **Presenting Necessary Information:** Only include information that helps achieve the audience's objectives. Avoid overwhelming users with unnecessary data.
- **Creating a Logical Workflow:** Ensure that users can easily navigate through the visualization. For instance, if a user selects an option, they should be directed to relevant information, much like calling a store's department without getting routed to the wrong area.
- **Utilizing Usability Studies:** Use insights from usability studies to inform your technical designs, allowing you to create a seamless user experience.

## Prioritizing Information

It's essential to prioritize the information presented to your audience. For example, if you discover a new feature in the dashboard system that doesn't add significant value, it's better not to implement it. Unnecessary features can distract users rather than enhance their experience.

## Conclusion

In summary, the effectiveness of your data visualizations hinges on thoughtful design decisions that prioritize the audience's needs. By paying attention to detail and organizing information logically, you can effectively communicate valuable

insights and keep users engaged. Remember, a well-designed visualization not only captures attention but also fosters ongoing interest in the data presented.

## ▼ Additional Design Concepts to Consider for Data Visualizations

Hello, and welcome to this discussion on transforming great data into impactful visualizations! As a cloud data professional, effectively communicating data insights is one of your core responsibilities. Let's explore several crucial factors that can influence your design decisions.

### 1. Accessibility of the Solution

When designing your visualizations, the first step is to consider how users will access the solution. Key questions include:

- **Device Usage:** Will users primarily be on laptops, desktops, mobile devices, or a combination of these?
  - For mobile users, it's important to recognize the implications of a limited display size. Too many visualizations can be overwhelming on a small screen, so prioritize the most critical visualizations that can fit comfortably.
  - Consider providing access to additional visualizations through links, tooltips, popups, or menus, ensuring users can easily explore more data without cluttering the interface.

### 2. Collecting User Feedback

Once your solution is implemented, gathering user feedback is essential for ongoing improvement. Here are important considerations:

- **Feedback Collection Methods:** Decide how you will collect feedback. Options include:
  - A feedback button linking to an online form or an email pop-up.
- **Type of Feedback:** Determine the nature of feedback you wish to receive:
  - Are you focusing solely on technical issues, or are you also open to suggestions for new features and enhancements?
- **Streamlined Process:** Make the feedback process easy and efficient for users, encouraging them to share their thoughts and experiences.

### 3. Aim for Intuitive Solutions

One of your primary goals as a data professional is to create a solution that is self-explanatory. Users should be able to navigate and understand the visualizations without needing additional instructions. However, it's also beneficial to provide:

- **Help Features:** Consider including help icons or tooltips to assist users as needed.
- **Clear Documentation:** Offer concise, accurate documentation that outlines how to use the visualizations and any other relevant information, ensuring users feel supported.

## 4. Storytelling through Visualizations

Effective data visualizations should convey meaningful stories and provoke thoughtful insights. When designed well, they provide significant value by transforming raw data into relevant, actionable insights for your organization.

### Conclusion

By considering these additional design concepts—accessibility, user feedback, intuitive solutions, and storytelling—you can create data visualizations that not only look good but also serve their purpose effectively. When your visualizations resonate with users and facilitate understanding, you play a vital role in harnessing your organization's data for impactful decision-making.

## ▼ Design Principles and Strategies for Dashboards

Attention to detail is a data professional's superpower. It involves spotting the little things that others might miss, like identifying errors in datasets or recognizing subtle differences in project plans. In this guide, you'll learn how to apply these principles to data visualization design, empowering you to become a data superhero! Here are several key strategies to enhance your dashboard design.

### 1. Prioritize Simplicity and Clarity

- **Central Design Goals:** Keep simplicity and clarity at the forefront of your dashboard design. Avoid cumbersome menus, layouts, and workflows that could confuse users.
- **Example:** Dashboards should visually display data in one place without unnecessary complexity.

### 2. Structure for Organization and Intuition

- **Logical Mapping:** Design a structure that is organized and intuitive. Ensure that visualizations are distributed evenly to avoid overcrowding.
- **Impact:** A cluttered dashboard can make it difficult for users to find the information they need.

### 3. Group Similar Data Types Together

- **Effective Visual Inferences:** Grouping similar data types can indicate relationships or connections between information, leading to better insights.
- **Visual Separation:** This approach also creates visual separations between unrelated data, providing users with essential context and meaning.

## 4. Prioritize Usability and Accessibility

- **Universal Access:** Ensure that everyone can use and understand your dashboards by applying usability and accessibility principles.
- **Consider Assistive Technologies:** Many users rely on screen readers and other assistive technologies, so make accessibility an integral part of your design.

## 5. Establish Consistent Use of Colors and Textures

- **Color Consistency:** Develop a clear plan for the consistent use of colors and textures throughout your dashboard. For instance, if one red visualization signals a "warning," ensure that all red elements convey the same meaning.
- **Label Consistency:** Labels should also maintain a consistent color scheme unless intentionally differentiating types or groups.

## 6. Be Intentional with Labels

- **Clear Communication:** Carefully construct labels to avoid confusion. Poorly worded labels can lead users to miss helpful information or misunderstand the visualizations.
- **Example:** Instead of naming a bar chart "Sales," use a clearer title such as "Year-over-Year Comparison" to provide better context.
- **Detail Definition:** Ensure axis labels, legends, and other references are clear. For example, clarify what y-axis numbers represent (e.g., dollars, units, thousands).

## Pro Tips for Consistency in Dashboard Design

1. **Document Terms and Definitions:** Include a legend that clearly defines all terms and definitions used in the dashboard.
2. **Uniformity Across Visualizations:** Ensure all visualizations adhere to the documented definitions for consistency.
3. **Clarify Exceptions:** If a deviation from established definitions is necessary, make sure to clearly note it.

4. **Standard Terminology:** Use the same terminology consistently across all parts of your solution, including department names, groups, and teams.

## Conclusion

By following these strategies and tips, you can become a true data superhero! Your attention to detail will be appreciated by dashboard users, demonstrating how much you value their needs and enhancing their experience with data visualizations. Remember, great dashboards not only present data effectively but also empower users to derive meaningful insights and make informed decisions.

## ▼ Methods for visualization planning and design

### ▼ Data Types to Consider for Visualizations

Welcome to this overview of data types! Understanding the different types of data is crucial for designing effective data visualizations and communicating insights clearly. In this guide, we'll explore the two basic types of data, their properties, and how they influence visualization choices.

## 1. Types of Data

- **Categorical Data (Qualitative Data):**
  - **Definition:** Categorical data describes qualities or characteristics and is subjective.
  - **Examples:** Countries, industries, products.
  - **Visualization:** Best represented in bar charts, column charts, and pie charts, which can illustrate relationships between categories.
- **Numerical Data (Quantitative Data):**
  - **Definition:** Numerical data represents specific, objective measures that can be counted, ordered, or measured.
  - **Examples:** Ages, test scores, prices.
  - **Visualization:** Suitable for line charts, histograms, scatter plots, box plots, and bubble charts, which effectively show changes over time, distributions, and values.

## 2. Common Data Visualization Types

There are five main ways to visualize data based on its type: single-value, comparison, composition, distribution, and relationship visualizations.

- **Single-Value Visualization:**
  - **Description:** Displays a single value derived from a query.

- **Example:** The average points per game of a football player.
- **Comparison Visualization:**
  - **Description:** Compares two or more values.
  - **Visual Types:** Bar charts, column charts, line charts.
  - **Example:** Comparing the population of the top 10 largest countries.
- **Composition Visualization:**
  - **Description:** Shows individual parts as a whole, typically summing to 100%.
  - **Visual Types:** Pie charts, stacked bar charts.
  - **Example:** An age distribution chart showing the percentage of people in each age group.
- **Distribution Visualization:**
  - **Description:** Illustrates how data correlates among different data points.
  - **Purpose:** Displays differences in averages, means, medians, etc.
  - **Example:** Showing the distribution of test scores across different student groups.
- **Relationship Visualization:**
  - **Description:** Demonstrates correlation between two or more variables.
  - **Visual Types:** Scatter plots, bubble charts.
  - **Example:** A bubble chart depicting growth rate versus median age.

### 3. Multi-Function Graphs

Some graph types can serve different purposes depending on how the data is interpreted. For instance, a line chart can be used for both comparison and distribution, depending on the context of the data.

### Conclusion

With numerous programs available for designing charts and graphs, you have various tools at your disposal for creating effective data visualizations. Always consider the type of data you are working with first to determine which visualization will be most effective for your audience. By understanding the nuances of categorical and numerical data, you'll be better equipped to communicate meaningful insights through your visualizations.

## ▼ Introduction to Business Intelligence Dashboards

Business intelligence (BI) dashboards play a crucial role in data analysis and decision-making across various organizational levels. Just as the mode of transportation chosen depends on specific needs, the type of dashboard you design must align with your users' requirements and how they intend to utilize the information. This guide outlines the four basic types of BI dashboards and key considerations for their design.

## 1. Types of Business Intelligence Dashboards

- **Strategic Dashboards:**

- **Overview:** These are high-level visualizations designed for corporate executives.
- **Purpose:** Focus on long-term organizational strategies and key performance indicators (KPIs).
- **Typical Displays:** Corporate financial performance and revenue trends.

- **Operational Dashboards:**

- **Overview:** More detailed dashboards suited for junior-level decision-makers and their teams.
- **Purpose:** Concerned with short-term operational processes and performance monitoring.
- **Typical Displays:** Business metrics such as sales activity, marketing performance, or customer support status.

- **Analytical Dashboards:**

- **Overview:** These dashboards assimilate large amounts of data for historical analysis.
- **Purpose:** Identify trends, make comparisons, create predictions, and set future goals.
- **Typical Users:** Middle managers or data analysts focused on ecommerce, sales, or website analytics.

- **Tactical Dashboards:**

- **Overview:** Highly detailed dashboards used for specific analysis, such as social media ads or sales manager KPIs.
- **Purpose:** Track initiatives and performance related to specific business areas.
- **Typical Users:** Stakeholders involved in detailed performance tracking.

## 2. Key Considerations for Dashboard Design

- **Data Sources:**
  - All dashboard types may utilize the same underlying data sources, but they present the data differently based on user needs.
- **User Groups:**
  - Certain user groups might require access to multiple dashboard types to address various questions and changing business contexts over time.
- **Flexibility:**
  - Different dashboard types can be adapted for multiple purposes, allowing for a blend of visualizations tailored to user roles and business goals.

## 3. Importance of Understanding Dashboard Types

As a cloud data analyst, recognizing the various types of dashboards and their specific uses is vital. Just like selecting the appropriate transportation method for different situations, the dashboard you design should be driven by the needs of your users and the objectives of your organization.

### Conclusion

In summary, effective business intelligence dashboards are essential tools for data-driven decision-making. By understanding the different dashboard types and tailoring them to user needs, you can enhance your organization's analytical capabilities and support informed decision-making. With this knowledge, you'll be equipped to create impactful dashboards that deliver significant value to your stakeholders.

## ▼ Benefits of Scorecards

Scorecards are essential tools in both sports and business environments, serving to track progress and measure achievements against established goals. Just as keeping score in games fosters competition and motivation, scorecards in a business context offer several key benefits.

### 1. Definition and Purpose of Scorecards

- **Scorecard Overview:** A scorecard is a statistical record that measures progress towards specific objectives. It allows stakeholders to compare multiple data points, facilitating informed decision-making.
- **Monitoring vs. Tracking:** Unlike dashboards, which primarily monitor progress, scorecards specifically track the metrics related to business goals.

### 2. Quick Identification of Performance

- **Spotting Actual Results:** Scorecards enable users to easily visualize actual performance results and compare them against pre-set goals.
- **Timely Decision-Making:** This comparison allows users to quickly assess whether action is needed to stay on track toward achieving objectives.

### **3. Contextual Understanding of Data**

- **Detailed Analysis:** For instance, a data analyst working for a call center can create a scorecard that reflects the relationship between sales goals and actual sales performance.
- **Informed Adjustments:** If sales are below target, the scorecard provides clarity on necessary adjustments to improve performance.

### **4. Enhanced Insights for Action**

- **Comprehensive Metrics:** Scorecards can display various metrics, such as first-time callers versus repeat callers or conversion rates, alongside departmental goals.
- **Identifying Trends:** By comparing current performance against desired outcomes, management can identify trends and issues, facilitating prompt responses to potential problems.

### **5. Proactive Problem-Solving**

- **Immediate Actions:** For example, if hold times increase due to a rise in call volume, management can hire additional staff to address the situation effectively.
- **Dynamic Response:** Scorecards transform data into actionable insights, enabling organizations to react dynamically to changing conditions.

### **6. Motivational Tool**

- **Engagement and Competition:** Just like in sports, knowing the score motivates teams to perform at their best. Scorecards provide visibility into performance, fostering a competitive spirit and accountability among team members.

### **7. Integration with Dashboards**

- **Complementary Insights:** While dashboards provide a broad overview, incorporating scorecards adds depth and context to the data visualizations.
- **Enhanced Understanding:** The combination of both tools enriches the analytical landscape, ensuring stakeholders have the necessary insights to drive performance improvements.

## Conclusion

In summary, scorecards play a vital role in measuring and improving business performance. By providing clear, comparative insights into metrics and goals, they empower organizations to make informed decisions, identify areas for improvement, and take proactive actions. Ultimately, scorecards serve as a bridge between raw data and strategic action, enhancing overall business effectiveness.

## ▼ SMART Questions for Stakeholders

Asking effective questions is a skill that can significantly enhance your work as a data professional. Drawing inspiration from influential figures like Socrates and Marie Curie, you can improve your inquiry methods by utilizing the SMART goal framework. This methodology ensures that your questions are specific, measurable, action-oriented, relevant, and time-bound.

### 1. Specific Questions

- **Importance:** Specific questions provide clarity and focus, prompting stakeholders to give detailed answers.
- **Example:** Instead of asking, "What kinds of employees will be using this system?", refine it to, "Will this solution be used by entry-level teams, project teams, or middle managers?"
- **Benefit:** This approach encourages stakeholders to think critically and provide information that is directly useful.

### 2. Measurable Questions

- **Importance:** Measurable questions help define clear, quantifiable objectives.
- **Example:** Instead of a vague inquiry like, "Do you want to increase sales?", ask, "What are your current sales figures, and by how much do you want to increase that?"
- **Benefit:** This specificity leads to actionable metrics that can be tracked over time.

### 3. Action-Oriented Questions

- **Importance:** Action-oriented questions stimulate discussion and elicit responses that can be acted upon.
- **Example:** Instead of asking, "Will you use market and industry benchmarks?", try, "Which market and industry benchmarks will you use?"
- **Benefit:** This encourages stakeholders to elaborate, fostering a deeper dialogue and generating valuable insights.

## 4. Relevant Questions

- **Importance:** Questions must align with the project's goals and business objectives.
- **Example:** Consistently ask questions like, "What problem are you trying to solve?" and "Which key performance indicators will you use to measure success?"
- **Benefit:** Relevant questions keep the conversation focused on the objectives at hand and ensure that the discussion remains meaningful.

## 5. Time-Bound Questions

- **Importance:** Specifying a time frame narrows the scope of analysis, ensuring focus on pertinent data.
- **Example:** Ask, "Should we analyze data from the past quarter or the last two quarters?" or "What is the timeframe for this project?"
- **Benefit:** Time-bound questions clarify deadlines and help prioritize tasks effectively.

## Setting Realistic Goals and Expectations

- After asking your SMART questions, it's essential to establish realistic goals. If stakeholders request an overly complex dashboard with every data point, this may not be feasible due to time and space constraints.
- **Outcome:** By setting achievable expectations, you are more likely to meet your goals and deliver valuable insights to stakeholders.

## Conclusion

Asking SMART questions is a powerful strategy that enhances your effectiveness as a cloud data professional. By fostering open dialogue and eliciting detailed responses, you can gain insights that lead to problem-solving and innovation. Ultimately, this approach can lead to remarkable discoveries and significant contributions in your field.

## ▼ Strategies for Translating Stakeholder Requests

As a data analyst, the concept of "translation" extends beyond language; it involves converting stakeholder requests into actionable data requirements. Here's how to effectively translate stakeholder needs to create valuable visualizations such as reports and dashboards.

### 1. Understand Your Stakeholders

- **Importance:** Building a rapport with stakeholders helps you comprehend their needs better.
- **Action Steps:**
  - **Know Their Background:** Learn about their areas of expertise and experience.
  - **Identify Communication Styles:** Determine their preferred communication methods and personality types.
  - **Gauge Comfort Levels:** Understand how comfortable they are discussing data-related topics.

## 2. Ask Detailed Questions

- **Importance:** Detailed questions help clarify stakeholder requirements and uncover deeper insights.
- **Action Steps:**
  - Use SMART questions (Specific, Measurable, Action-oriented, Relevant, Time-bound) to guide your inquiries.
  - Example: Instead of accepting a request for a report with a specific number of columns, ask, "What specific insights do you want to gain from these columns?"

## 3. Explore the "Why"

- **Importance:** Understanding the rationale behind a request can reveal additional needs and expectations.
- **Action Steps:**
  - Ask follow-up questions that encourage stakeholders to elaborate on their needs.
  - Example: "Why is it important to have this data presented in this way?"

## 4. Continue the Dialogue

- **Importance:** Ongoing communication is vital for capturing the full scope of the project.
- **Action Steps:**
  - Keep the conversation open and iterative, revisiting key points as new information emerges.

- Encourage stakeholders to share any concerns or additional thoughts they may have.

## 5. Synthesize Information

- **Importance:** Once you gather enough details, you need to consolidate the information to clarify requirements.
- **Action Steps:**
  - Summarize the key points discussed and ensure they align with the stakeholder's goals.
  - Present a draft of your understanding back to the stakeholder for confirmation.

## 6. Deliver Concise Solutions

- **Importance:** Effective translation results in solutions that meet stakeholder needs succinctly and clearly.
- **Action Steps:**
  - Develop reports and dashboards that are tailored to the insights and metrics the stakeholders are interested in.
  - Use visualizations that are easy to interpret, keeping the end-user in mind.

## Conclusion

Translating stakeholder requests is a crucial skill for data analysts. By understanding stakeholders, asking insightful questions, and maintaining open communication, you can effectively translate their needs into actionable data. This approach not only enhances your effectiveness in your role but also strengthens collaboration with stakeholders, ultimately leading to better decision-making and business outcomes.

## ▼ Introduction to Wireframes for Stakeholder Alignment

Creating initial drafts is a crucial step in the development process for various professionals. Just as architects sketch building designs and graphic designers create logo mockups, data professionals can leverage wireframes to visualize their ideas before executing a project. Here's how wireframes can facilitate effective stakeholder alignment and improve project outcomes.

### 1. What is a Wireframe?

- **Definition:** A wireframe is a visual representation that outlines the structure and functionality of a user interface or product layout. In the context of dashboards, it illustrates the basic layout and design before the final implementation.
- **Purpose:** Wireframes serve as prototypes that allow creators to communicate their vision clearly, obtaining feedback before investing significant time and resources into a project.

## 2. Importance of Wireframing in the Development Process

- **Stakeholder Alignment:**
  - Wireframes provide a visual format that stakeholders can easily understand, ensuring everyone is on the same page regarding the project's direction.
  - This clarity increases the likelihood of developing a final product that meets stakeholder expectations.
- **Simplifying the Feedback Process:**
  - Visual drafts make it easier for stakeholders to review and provide feedback on the proposed design.
  - Quick reviews lead to timely adjustments, streamlining the development process.

## 3. Benefits of Using Wireframes

- **Enhanced Communication:**
  - Wireframes convert complex ideas into straightforward visuals, making discussions more productive.
  - Stakeholders can visually assess design choices and suggest changes without misunderstanding verbal or written descriptions.
- **Facilitating Accessibility:**
  - Wireframes help improve the accessibility of your design by allowing you to arrange elements like navigation, images, and text effectively.
  - This focus ensures users can easily interpret data, accommodating diverse user abilities.

## 4. Tools for Creating Wireframes

- **Dedicated Wireframe Programs:**
  - Various software tools are available specifically for wireframing that offer features tailored for user interface design, helping to illustrate how different

elements interact within the dashboard.

- **Alternative Applications:**

- If access to wireframing software is limited, you can still create visual drafts using general applications like Google Slides or Google Docs.
- While these tools may lack advanced features, they allow for basic visual documentation of your layout and planning.

## 5. Conclusion

Wireframes play a vital role in aligning stakeholder needs and simplifying the feedback process in project development. By utilizing wireframes during the planning stages, data professionals can gain valuable insights from stakeholders, saving time and effort while uncovering design challenges and opportunities for improvement. Whether you choose a specialized wireframe program or a basic application, the goal remains the same: to create a clear, actionable design that meets the expectations of your stakeholders and enhances the overall user experience.

## ▼ Access, explore, and report on data in the cloud

### ▼ The Data Journey: Analyze and Activate

The data journey consists of five essential stages: collect, process, store, analyze, and activate. This segment focuses on the last two stages—**analyze** and **activate**—which are crucial for transforming raw data into actionable insights.

#### 1. Analyze Stage

- **Objective:** The primary goal during the analyze stage is to identify trends and patterns in your data, enabling you to uncover insights that are valuable to your users.
- **Tools:** Cloud-based data visualization tools offer a range of features that facilitate data exploration and analysis:
  - **Filtering Data:** Allows you to narrow down data sets to focus on specific information.
  - **Drilling Down:** Enables you to explore data in greater detail, revealing granular insights.
  - **Custom Visualizations:** Create tailored visual representations of your data that highlight key trends.
- **Outcome:** By leveraging these features, data professionals can develop deeper insights and a better understanding of the data, leading to more informed decision-making.

## 2. Activate Stage

- **Objective:** The activate stage involves presenting your visualizations to stakeholders and using the insights gained from the data to inform decisions and prompt action.
- **Communication:**
  - Visualizations serve as a clear and concise way to convey insights, making it easier for teams to understand and act on the information.
  - Sharing visualizations created with cloud-based tools enhances collaborative decision-making across teams.

## 3. Practical Example: Analyze and Activate in Action

- **Scenario:** As a data analyst, you might analyze data regarding national sneaker sales trends alongside your company's local sales data.
- **Visualization:** By combining these datasets into a single graph, you can provide a comparative view of regional sales against national averages.
- **Impact:** The sales team can utilize this visual representation to develop more effective sales strategies, tailoring their approach based on the insights gained.

## 4. Benefits of Cloud-Based Data Visualization Tools

- **Flexibility and User-Friendliness:** Cloud-based tools allow data professionals to analyze data from various sources, including local databases, spreadsheets, public datasets, and cloud-based databases.
- **Time Efficiency:** These tools significantly reduce the time spent on manual processing and analysis, enabling quicker insights and decisions.
- **Empowerment:** Users can interact with and explore the data intuitively, encouraging them to ask deeper questions and make data-driven decisions.

## 5. Conclusion

Utilizing cloud-based data visualization tools to analyze and activate data not only enhances your ability to derive meaningful insights but also fosters collaboration and informed decision-making within your teams. By discovering, accessing, and visualizing data effectively, you can unlock new opportunities for both yourself and your organization, transforming how data is leveraged in strategic planning and execution.

## ▼ Considerations for Connecting to Cloud Data

The cloud data profession offers exciting opportunities due to the vast amounts of data accessible from various sources. This video will discuss different types of

data sources and key considerations for connecting to cloud data.

## 1. Types of Data Sources

As a cloud data analyst, you will likely encounter four common types of data sources:

- **Public Data:**

- Collected from businesses, government entities, and private organizations.
- Publicly available data can advance various sectors, including education, social causes, economic growth, and innovation.

- **Product-Specific Data:**

- Available across various industries such as retail, non-profit, and finance.
- Accessed through data management systems to gain insights specific to particular products or services.

- **Software Platform Data:**

- Encompasses data from user relationship management (CRM), social media, advertising, human resources, and financial systems.
- These platforms can provide valuable insights into customer interactions and business operations.

- **Company-Specific Sources:**

- Includes systems like Salesforce, proprietary databases, and spreadsheets.
- These sources often have unique formats, which may affect how data is visualized and analyzed.

## 2. Connecting to Data Sources

- As a cloud data analyst, you have various methods to connect to the aforementioned data sources.
- Tools like Looker and Looker Studio can help you utilize internet-accessible data efficiently.

## 3. Key Considerations When Connecting to Cloud Data

- **Data Freshness:**

- Refers to how current the data is in reports or dashboards.
- For instance, if a dashboard is updated every 24 hours, data accessed later could be outdated, impacting decision-making.

- The importance of data freshness can vary depending on the business needs; for example, an executive dashboard might not require frequent updates, while a sales team dashboard likely does.
- **Data Security:**
  - Awareness of user permission groups is crucial for maintaining security. Typical permission groups include:
    - **Admin:** Highest level with full access to manage workspaces, assign roles, and update databases.
    - **Developer:** Works with data modeling languages to create models and manage dashboards.
    - **User:** Explores data and creates visualizations and custom fields.
    - **Viewer:** Has the most limited access, allowing them only to interact with pre-existing visualizations.
  - Understanding these roles and reviewing documentation for each tool is essential for effective data management.
- **User Permissions:**
  - Be aware of how and when users can access data.
  - Protecting company data is crucial, especially from unauthorized access, whether users are on mobile devices or laptops in different environments.

## 4. Conclusion

As a cloud data professional, it is vital to consider these factors when accessing data from a variety of sources. The availability of data from anywhere at any time is remarkable, but it's your responsibility to ensure its use is responsible and secure. By addressing these considerations, you can effectively create dashboards that leverage both public and company-specific data to support insightful decision-making for stakeholders.

## ▼ The Importance of Data Exploration

When faced with something unfamiliar, such as a new place, we often begin by identifying its details and characteristics to better understand it. This exploratory process is vital for decision-making and problem-solving, and it is equally important in data analytics. In this video, we will explore the concept of data exploration and its significance.

### 1. What is Data Exploration?

As a cloud data analyst, data exploration involves understanding a dataset by examining its characteristics, identifying patterns, and posing questions. Key attributes to consider include:

- **Size:** The total number of records in the dataset.
- **Quantity:** The amount of data across various dimensions.
- **Distribution:** How values are spread across the dataset.
- **Accuracy:** The reliability of the data.

Understanding these attributes helps analysts grasp the essential nature of the data they are working with.

## 2. Importance of Data Exploration

- **Understanding Dataset Structure:**
  - Grasping the structure of a dataset is fundamental. It serves as a foundation for recognizing the size, quantity, and type of data involved.
  - Think of data exploration like navigating a filing cabinet, where labels on folders help locate specific files. If folders are poorly labeled, finding the right information becomes challenging.
- **Defining Data Columns:**
  - Accurate labeling of data columns is crucial for clarity. Similar or vague labels can lead to confusion and errors in data analysis.
  - Data exploration allows analysts to "open the folders" and comprehend the content of each data column.
- **Experimenting with Visualizations:**
  - Data exploration is an opportunity to experiment with different ways to visualize data, refining visualization plans for better clarity.
  - Analysts can tailor their questions and hypotheses during the exploration process, leading to more effective data analysis.
- **Confirming Data Quality:**
  - Exploring data enables analysts to verify data quality and the distribution of values.
  - Identifying outliers and data errors is essential for ensuring accurate visualizations and informed decision-making.

## 3. Tools for Data Exploration

There are various tools available for data exploration:

- **Coding Languages:**
  - Python, SQL, and R are powerful languages for exploring and manipulating data.
- **Spreadsheet Programs:**
  - For smaller datasets, tools like Google Sheets or Excel can be effective for exploration.
- **Data Visualization Tools:**
  - Visual tools like Looker can assist in exploring data relationships, allowing for quick identification of potential issues before they escalate.

## 4. Conclusion

Mastering data exploration is essential for becoming a highly effective data professional. It equips you with the skills to understand new datasets thoroughly and to identify issues early in the analysis process. Thank you for joining this exploration of the importance of data exploration!

## ▼ Dimensions and Measures in Data Models

Today, I'm excited to guide you through the essential concepts of data models, specifically focusing on dimensions and measures, and their value in data analytics.

### Understanding Data Models

A **data model** is a framework for organizing data elements and understanding how they relate to one another. It can be visually represented, illustrating the structure and relationships within the data.

- Every data source possesses a data model, whether it's explicitly defined or not.
- For instance, a simple table consists of rows and columns that define the data model. In contrast, a more complex data structure, like a snowflake schema, involves intricate relationships between different tables.

As data analysts, creating data models is crucial for structuring datasets and defining data attributes.

### The Role of Dimensions and Measures

**Dimensions** and **measures** are fundamental components of data attributes that allow for flexible exploration of datasets, enabling analysts to ask meaningful questions about the data.

## What Are Dimensions?

- **Dimensions** are unique attributes of data that help describe it.
- For example, in a data model for a bookstore, dimensions might include:
  - Title
  - Genre
  - Price
  - Date Published
  - In Stock (availability)

Each row in the table represents a single book and includes values for each dimension, structured in columns.

- Dimensions help group values together, allowing users to ask insightful questions. For instance, a bookstore owner could query which genres are available in their inventory by utilizing the "genre" dimension.
- Moreover, dimensions can be combined for more complex inquiries. For example:
  - "What genres are available, and which of the books within each genre are currently in stock?" This question requires using both the "genre" and "in stock" dimensions.

## What Are Measures?

- **Measures** are aggregations derived from one or more dimensions, such as counting or averaging values.
- Using the bookstore example, if you want to know how many books are in total, you need to apply a measure—specifically, the count.
- In this context, each row represents an individual book. To calculate the total number of books in inventory, you would count the rows, potentially focusing on a unique identifier column to determine the total number.
- Measures can also be utilized to calculate other aggregates, such as the total price of all books or the average price of books in inventory.

## Significance of Dimensions and Measures in Data Analytics

Dimensions and measures are vital in data analytics, particularly when it comes to data visualization. They provide crucial insights into data relationships and enhance the overall understanding of the dataset.

- By investigating and leveraging dimensions and measures in your data models, you can maximize the potential of any dataset you encounter.

Thank you for joining me on this exploration of dimensions and measures! Keep experimenting with these concepts to enrich your data analysis skills and uncover valuable insights.

## ▼ Basics of Modeling Data

In the world of data analytics, preparing your data is a crucial step before it's ready for reports and dashboards. This preparation process is known as **data modeling**.

### Why Model Data?

There are several key reasons for modeling data:

1. **Combining Multiple Data Sources:** If you're working with different data sources, data modeling helps integrate and prepare them for analysis.
2. **Understanding Data Better:** It allows you to design, structure, join, and transform data, making it easier to work with and comprehend.

Data modeling involves manipulating the data to create a clearer and more organized structure. This may include defining joins between datasets or standardizing column values.

### Data Modeling Techniques

Here are some common techniques used in data modeling:

#### 1. Filtering:

- **Filtering** is the process of displaying only the data that meets specific criteria while hiding the rest.
- You can either include data that matches certain conditions or exclude data that doesn't.
- **Example:** In a movie database, you might filter to show only comedies (include) or display everything except horror films (exclude).

#### 2. Blending Data:

- **Data blending** combines data from multiple sources to create a single report or visualization.
- This enriches your dataset and enhances flexibility.
- Note: Blends aren't reusable across reports since they involve various sources, so you should be aware of this limitation.

#### 3. Aggregations:

- **Aggregations** summarize data through functions like sum, average, count, minimum, and maximum.
- These functions alter how data is displayed in visualizations and can be adjusted as needed.
- Aggregations help in grouping and summarizing data effectively.

## Adjustments and Modifications

While modeling data, you might encounter situations where the data doesn't appear as intended. Here are some considerations:

- **Modifying Column Titles:** If a column title is unclear, update it to be more descriptive. You can do this for a single visualization or change it in the data source for universal clarity.
- **Calculations:** You can adjust calculations for specific visualizations, like setting a column to sum or average. Alternatively, you can make changes in the data source to apply universally for future models.

## Conclusion

Modeling data is essential for gaining deeper insights into any dataset. By mastering data modeling techniques, you'll be better equipped to make informed decisions regarding data visualization and effectively share valuable insights.

Keep learning and exploring the world of data modeling to enhance your analytical skills!

## ▼ Data Blending for Helpful Insights

In this session, you'll discover how data from different sources can be combined to provide valuable insights for your analyses.

### What is Data Blending?

Data blending is the process of combining data from multiple sources to create a single visualization. By merging, aggregating, and connecting data relationships from separate tables, you create a fully integrated dataset that is much more powerful.

- **Why Blend Data?**
  - By putting disparate data together, you can generate visualizations that enhance your reports, providing users with additional guidance for analysis and decision-making.
  - For instance, a company might merge data from its local database, a public cloud source, and a software application into one effective visualization.

- This approach makes data more accessible and broadens the scope of insights, allowing users to have deeper, more informed analyses.

## Key Steps in Data Blending

Let's explore some essential steps and considerations for effective data blending:

### 1. Understanding the Differences:

- While blending data may seem similar to creating a dataset, there are key differences. Blends are embedded within the report they were created in, meaning they are only usable within that specific report.
- To reuse a blend, you can copy the report, which will include the blend along with it.

### 2. Data Freshness:

- The freshness of the blended data relies entirely on the underlying data sources. The blended dataset will be as current as its source data.

### 3. Using Business Intelligence Tools:

- Various business intelligence (BI) tools facilitate quick and easy data blending. The general process typically includes:
  - **Connecting to Data Sources:** Most programs can combine more than two data sources.
  - **Defining Joins:** Join data to create a single dataset. Ensure that each dataset has at least one common column to establish essential relationships that the BI platform can understand.
  - **Cleaning Data:** Remove unnecessary data, correct data entry errors, and format the data optimally. This preparation is crucial for effective analysis and visualization.

## Benefits of Data Blending

Blending data provides users with a comprehensive view, enabling the identification of patterns and trends. Here are some of the benefits:

- **Greater Insights:** Users can gain deeper insights into their data.
- **Accessibility:** Insights become more accessible, making it easier for users to analyze and make informed decisions.
- **Enhanced Visualizations:** Effective graphs, charts, tables, and single-value visualizations can be generated, enhancing the overall reporting experience.

## Conclusion

Data blending is a powerful tool for data analysts, allowing for a more nuanced understanding of data and fostering insightful analyses. By combining different data sources effectively, you can uncover patterns, trends, and insights that may otherwise remain hidden.

Keep exploring the potential of data blending to unlock valuable insights in your analyses!

## ▼ Introduction to Data Reports

As a data analyst, you'll frequently utilize both **reports** and **dashboards** when developing data visualizations. These two forms of visualization are essential tools in your toolkit.

### What are Data Reports?

In this section, we'll delve into what data reports are, how they are used, their benefits, and the fundamental differences between reports and dashboards.

- **Understanding Data Reports:**

- While you might think you know what reports are, their application in data analysis may differ from your expectations.
- Unlike text-based reports, such as news articles or scientific publications, a **data report** visualizes detailed business intelligence data to aid in making informed business decisions.
- These reports visually present data insights aimed at facilitating action regarding business needs or objectives. They can take various forms, including:
  - Research reports presenting findings of studies
  - Progress reports detailing project statuses
  - Financial reports highlighting business performance

### Similarities with Dashboards

Let's explore the similarities between reports and dashboards:

- **Visualization of Data:** Both reports and dashboards are methods of visualizing and activating data.
- **Common Tools:** They can be built with similar tools and may even appear alike.
- **Multiple Sources:** Both can utilize multiple data sources to present insights.
- **Accessibility:** They enhance data accessibility for users, enabling decision-making, collaboration, and sharing of critical insights.

## Key Differences Between Reports and Dashboards

While reports and dashboards share similarities, there are crucial differences:

### 1. Nature of Representation:

- Reports are typically **curated** and **static** representations. Once generated, the data in a report doesn't change, making it a snapshot of a specific point in time. In contrast, dashboards may be updated in near-real-time.

### 2. Detail and Length:

- Reports contain more detail and are generally longer than dashboards. They offer extensive content and explanations, while dashboards are designed for quick, at-a-glance understanding without additional context.

### 3. Time Sensitivity:

- Reports have an expiration date regarding their effectiveness. For instance, in a large hotel construction project, a report on construction costs would only be relevant for a specific time. In contrast, dashboards provide ongoing insights.

### 4. User Experience:

- The length and detail of reports may require users to spend more time reviewing and comprehending the content, unlike dashboards, which are meant for quick insights.

## When to Use Reports vs. Dashboards

Both data reports and dashboards serve unique purposes in data analytics:

- Data Reports:** Ideal for collaborating and answering specific business questions that require detailed insights, especially for one-time inquiries.
- Dashboards:** Best suited for repeated, ongoing, up-to-date data insights.

## Conclusion

In summary, understanding the distinctions and use cases for reports and dashboards is crucial for effective data analysis. Prioritizing user needs will help you determine which type of visualization works best in each situation, ensuring you provide the most valuable insights.

## ▼ Enterprise business analytics

### ▼ Introduction to Enterprise-Grade Visualization Tools

Enterprise-grade visualization tools are essential for businesses of all sizes to effectively manage and interpret large volumes of data. In this video, we will explore what these tools are, their features, and how they can be utilized.

## What Are Enterprise-Grade Visualization Tools?

An **enterprise-grade data visualization tool** is software designed for large, data-driven organizations to explore, analyze, and share business analytics. These tools can securely handle vast amounts of data, enabling users of all skill levels to work effectively with data.

## Key Features of Enterprise-Grade Visualization Tools

Let's delve into the common features of these tools:

### 1. Performance Optimization:

- Performance optimization is critical for organizations dealing with high volumes of data. Users need quick access to information and the ability to view reports in near-real-time.
- Enterprise-grade visualization tools include advanced features that allow for efficient access and interpretation of large datasets.

### 2. Metadata Management:

- **Metadata** is essentially data about data, providing descriptions and context for data resources.
- Examples include dataset names, descriptions, user permissions, and change histories.
- **Metadata management** involves organizing and accessing this metadata, which is essential for quick data discovery and effective collaboration.

### 3. Data Cataloging:

- A **data catalog** serves as a centralized inventory of an organization's data assets, collecting metadata about these assets.
- It includes information such as the meaning, origin, quality, and storage location of the data.
- Data analysts can track and manage data assets through catalogs, ensuring compliance with data policies.

### 4. Organization-Wide Metrics:

- These metrics are defined and shared across teams, often used to establish **Key Performance Indicators (KPIs)** like revenue and user satisfaction.
- Defining organization-wide metrics ensures that everyone is using the same indicators to measure impact, fostering a consistent data language throughout the organization.

## 5. Self-Service Analytics:

- Self-service analytics empower users to work with data independently, enhancing collaboration and enabling teams to maximize the value of their data.

## 6. Data Governance:

- As more users engage directly with data, **data governance** becomes crucial for safeguarding sensitive information and ensuring that users only access the data they require.
- This process involves the formal management of a company's data assets, which is essential for maintaining data security and efficiency as users interact and communicate about data.

## Conclusion

In summary, enterprise-grade data visualization tools provide a range of features that facilitate secure and efficient data interpretation across organizations. As you advance in your role as a cloud data analyst, you'll be equipped to recommend these tools when appropriate, helping businesses leverage their data effectively.

## ▼ Comparison of Self-Service and Guided Analytics

### Understanding Self-Service Analytics

**Self-service analytics** is a business intelligence approach that enables both technical and non-technical users to access data, perform ad-hoc analyses, and generate reports independently. This method caters to the growing demand for quick insights from various users within the organization.

### Understanding Guided Analytics

In contrast, **guided analytics** involves a more traditional model where analysts or developers create predefined solutions—such as reports and dashboards—to meet specific business needs. This approach provides users with tailored insights but relies heavily on the data team to deliver these solutions.

### Key Differences and Scenarios

To illustrate the differences, consider a scenario where you're hiring a tour guide in Paris:

- **Guided Analytics:** Similar to having a tour guide who plans an itinerary, guided analytics offers a structured path through data. The user identifies a need, and the data team develops a solution to guide decision-making. This method

ensures that users receive actionable insights, while the data team can address issues of data governance and privacy more effectively.

- **Self-Service Analytics:** Imagine exploring Paris on your own with a map. Self-service analytics gives users direct access to datasets, allowing them to create visualizations and run reports independently. This approach enhances user control over data analytics and speeds up decision-making.

## Advantages and Challenges

### 1. Guided Analytics:

- **Advantages:**
  - Provides predefined tools and insights, easing the user's experience.
  - Maintains data governance and privacy through oversight from the data team.
- **Challenges:**
  - Strains resources of the data team with numerous requests.
  - Delays in access to insights may hinder timely decision-making.

### 2. Self-Service Analytics:

- **Advantages:**
  - Empowers users to analyze data directly and make quicker decisions.
  - Increases engagement with data across teams.
- **Challenges:**
  - Users may lack awareness of broader data contexts, limiting their insights.
  - User-friendliness is crucial; a complicated system may deter adoption.

## Collaboration and Best Practices

To maximize the effectiveness of self-service analytics, it's vital for data teams to collaborate with users. This partnership helps users understand the data better, formulate relevant questions, and utilize self-service tools effectively.

## Conclusion

Both **self-service** and **guided analytics** offer unique advantages in promoting data-driven decision-making. As a cloud data analyst, your choice between these approaches will depend on the specific needs of your organization and its users, ensuring that data insights are accessible and actionable across all levels.

## ▼ Visualization Tools for Enterprise Data Exploration

Exploring data can often feel like putting together a huge puzzle—challenging yet rewarding. As you examine each piece, you begin to see patterns that help fit them together. This process of understanding a dataset is known as **data exploration**.

### What is Data Exploration?

Data exploration involves inspecting the characteristics of a dataset, identifying patterns, and asking questions. Using visualization tools during this process allows analysts to answer critical questions about the data effectively. Let's illustrate how this works through an example.

### Case Study: Arjun, the Data Analyst

Meet Arjun, a data analyst at a large clothing company. With a massive database at his disposal, Arjun must first understand the data's structure to analyze it effectively. To achieve this, he employs a few key techniques:

#### 1. Sampling:

- **Definition:** Sampling involves selecting a representative segment of a dataset to gain insights about its overall characteristics.
- **Benefit:** By using a smaller, manageable amount of data, Arjun can focus on specific questions and refine his analysis.

#### 2. Visual Exploration:

- Arjun then visualizes the sampled data using graphs like histograms and scatter plots.
- **Purpose:** These visualizations help him spot patterns, trends, and relationships within the data, which can lead to deeper insights.

#### 3. Identifying Outliers and Anomalies:

- Data visualizations also assist Arjun in identifying outliers and anomalies that might affect data quality.
- **Importance:** Addressing these issues before conducting in-depth analyses ensures the accuracy of the insights derived from the data.

#### 4. Data Drilling:

- To delve deeper into the dataset, Arjun uses a technique called **data drilling**.
- **Function:** This allows him to explore specific attributes of the dataset at a more granular level, fine-tuning his questions and queries.

## The Iterative Nature of Data Exploration

Arjun's journey emphasizes that data exploration is not always linear. Analysts often need to iterate through various steps, revisiting different techniques as they uncover more insights about the data.

## Key Takeaways

- **Iterative Process:** Data exploration is an iterative process that requires flexibility and adaptability based on the dataset and insights uncovered.
- **Critical First Step:** Engaging in thorough data exploration is crucial for the overall data visualization process. It helps analysts:
  - Understand the data better.
  - Save time and resources.
  - Ensure accurate, insightful analyses that meet data goals.

By effectively leveraging visualization tools during data exploration, analysts like Arjun can piece together the puzzle of their datasets, leading to more informed decision-making within the organization.

## ▼ Data Drilling: Up, Down, and Through

When capturing images with a camera, you have the ability to zoom in for close-ups or zoom out for a broader perspective. Similarly, **data drilling** allows you to explore data in detail by revealing additional levels of information. This process can be categorized into three main types: **drill down**, **drill up**, and **drill through**.

## Understanding Dimensional Hierarchies

Before delving into the specifics of each drilling technique, it's essential to grasp **dimensional hierarchies**. A dimensional hierarchy defines the levels of detail in a dataset that a chart can display.

- **Example:** For time, a dimensional hierarchy might include levels like:
  - Year
  - Month
  - Week
  - Day
  - Hour

Each level represents a different granularity of the data, with "year" being the most general and "hour" being the most detailed. The order of these levels is crucial as it dictates how users navigate through the data.

## Drilling Down

- **Definition:** Drilling down involves moving down the hierarchy from general to more granular data, revealing additional levels of detail.

## Example of Drilling Down

- **Scenario:** Sam, a data analyst at a community kitchen, looks at a chart displaying the total pounds of food donated each year.
- **Action:** To gain a more detailed view, Sam drills down to see the pounds of food donated by month.
- **Result:** The chart now shows monthly donations, revealing that December had the highest donations, followed by November and October, with noticeable dips during the summer months. This granular view helps Sam understand seasonal trends in food donations.

## Drilling Up

- **Definition:** Drilling up is the reverse process, where you move up the hierarchy from granular to more general data, resulting in fewer details.

## Example of Drilling Up

- **Scenario:** Sam reviews a chart showing weekly visitor counts at the community kitchen.
- **Action:** To see longer-term trends, Sam drills up to view the data for the entire month.
- **Result:** This provides a holistic view of the visitor data, allowing Sam to analyze trends over a broader time frame. Drilling up helps in understanding overall patterns rather than focusing on specific weeks.

## Drill Through

- **Definition:** Drill through allows users to navigate to related visualizations, providing a broader context across multiple charts.

## Example of Drill Through

- **Scenario:** Sam works with a visualization that displays various types of food donated by category over the last month. One category is cereal.
- **Action:** Curious about cereal usage, Sam clicks on the cereal label.
- **Result:** This action takes Sam to a new, pre-filtered report focused solely on cereal, allowing for quick access to relevant data and trends. Drill through

facilitates deeper exploration of specific data points.

## Conclusion

Drilling up, down, and through offers different perspectives on your data:

- **Drill Down:** For detailed insights into specific segments.
- **Drill Up:** For a broader understanding of overall trends.
- **Drill Through:** For navigation across related visualizations.

The best technique to employ depends on your data context and business needs, enhancing your ability to derive meaningful insights from your datasets.

## ▼ Live Dashboard Features

When a traffic event occurs, traffic analysts must act quickly. They monitor a variety of constantly updating indicators, including traffic cameras, GPS data, and weather reports, to predict traffic flow and guide commuters in planning their travels. A **live dashboard** serves as a data visualization tool, providing near-real-time updates. This capability is not limited to traffic monitoring; various industries utilize live dashboards to make quick decisions when timely information is critical.

### Key Features of Live Dashboards

Three essential features of live dashboards include **time-sensitive data**, **automatic refreshing**, and **alerting**. Let's explore each feature in detail.

#### 1. Time-Sensitive Data

- **Definition:** Also known as perishable data, time-sensitive data must be acted upon within a specific timeframe to retain its value.
- **Example:** A stock ticker illustrates this concept well, as it continually updates with the latest stock prices. Investors, traders, and brokers rely on this steady stream of information to identify trends and make decisions regarding buying, selling, or holding stocks.
- **Functionality:** Similar to a stock ticker, a live dashboard presents time-sensitive data through charts and graphs, enabling decision-makers to spot trends and respond to urgent business questions effectively.

#### 2. Automatic Refreshing

- **Definition:** This feature allows dashboards to update automatically at regular intervals, ensuring users always have a current overview of their metrics.
- **Benefits:** Automatic refreshing enables users to quickly assess the state of their data without manual intervention, streamlining their monitoring

process.

### 3. Alerting

- **Definition:** Since few teams can monitor dashboards continuously, alerting provides notifications when specific predetermined conditions are met or exceeded.
- **Importance:** This feature helps data teams detect issues early, even if team members are not actively observing the dashboard. Alerts keep everyone informed of significant changes in data, facilitating timely action.

These three features—time-sensitive data, automatic refreshing, and alerting—make live dashboards invaluable for monitoring urgent, time-sensitive data, such as key performance indicators (KPIs).

## Real-World Example

Let's illustrate how these features work in practice:

- **Scenario:** Joe, Zara, and Chang manage their business's webpage using a live dashboard to track website KPIs.
- **Observations:**
  - Joe notices a decrease in the average time visitors spend on each page.
  - Concurrently, Zara observes an increase in page load time.
- **Collaboration:** Recognizing the potential problem, Joe and Zara share their findings with Chang. Using the live dashboard, Chang can quickly compare the charts and confirm that both trends began around the same time, indicating a possible issue.
- **Investigation:** The team collaborates to investigate further, leveraging near-real-time data to identify trends and react swiftly to the situation.

## Handling Alerts

What if the team is unavailable when an issue arises? Alerts ensure they stay informed.

- **Setup:** The team configures the dashboard to send notifications when web traffic exceeds or falls below certain thresholds. These thresholds signal potential issues requiring immediate attention.
- **Proactive Response:** This setup enables the team to respond promptly to potential problems, even when not actively monitoring the dashboard.

## Conclusion

Live dashboards provide a range of applications, from tracking traffic patterns to monitoring website activity. By offering timely data updates, automatic refreshing, and alert notifications, live dashboards empower teams to make informed, time-sensitive decisions and stay ahead of trends. As a data analyst, you'll likely find live dashboards to be a powerful asset in your work, enhancing your ability to analyze and respond to critical data effectively.

## ▼ The Changing Role of a Cloud Data Analyst

### Introduction

Organizations now have access to vast amounts of data, fundamentally transforming how teams approach and utilize data for decision-making. Historically, data teams acted as gatekeepers, managing datasets and controlling how data was analyzed. However, the rise of self-service analytics and an emphasis on data-driven decision-making across various teams have led to significant changes in the data team's role.

### From Gatekeepers to Facilitators

Today, data teams are shifting from gatekeepers to facilitators. Their focus is on working cross-functionally to ensure the safe, uniform, and efficient use of data throughout the organization. This new role entails several responsibilities that enhance the overall effectiveness of data usage.

### Key Responsibilities

#### 1. Improving Data Literacy

- **Definition:** Data literacy refers to the ability to understand and use data effectively.
- **Impact:** With non-technical employees increasingly engaging with data, fostering data literacy has become crucial.
- **Role of Data Analysts:** As part of the data team, cloud data analysts may help define necessary skills for data literacy tailored to specific roles. They might also conduct training sessions to empower employees to utilize data insights effectively.

#### 2. Establishing Data Governance

- **Definition:** Data governance is the formal management of a company's data assets.
- **Importance:** As more team members gain direct access to data, robust data governance policies are essential to ensure data security and integrity.

- **Implementation:** Analysts might define and assign user roles and responsibilities, ensuring everyone has access to the data they need while keeping it secure.

### 3. Promoting a Common Data Language

- **Definition:** A common data language refers to standardized definitions and terms used to describe data across the organization.
- **Goal:** This ensures everyone has a unified understanding of data-related concepts, reducing ambiguity and improving data quality.
- **Role of Data Analysts:** Data teams may be responsible for optimizing and structuring data to support this common language.

### 4. Fostering a Data-Driven Culture

- **Definition:** A data-driven culture encourages collaboration between technical and non-technical employees, empowering them to make informed decisions using data.
- **Analyst's Role:** As cloud data analysts, individuals leverage their expertise to facilitate collaboration, encouraging teams to share data resources, work together on data-related projects, and drive decision-making.

## Conclusion

The role of a cloud data analyst is rapidly evolving. No longer seen as mere gatekeepers, data analysts now function as facilitators within organizations. They not only work directly with data but also collaborate across teams to cultivate a data-driven culture. This transformation empowers both technical and non-technical employees to make informed decisions with data, ultimately leading to more effective and agile organizations.

## ▼ Explore the developer environment

## ▼ Bringing Data Tools Together with an IDE

## Introduction

When tackling a project, having a user-friendly set of tools at your disposal can significantly enhance efficiency. For instance, consider writing a letter.

Traditionally, using a typewriter might require a dictionary for spelling and grammar checks, which can be tedious and error-prone. Today, word processing applications offer automated features that check grammar, spelling, and formatting, saving time and minimizing mistakes.

Similarly, an IDE consolidates the necessary tools for developers into one user-friendly environment, making the coding process more efficient.

## Using an IDE for Visualization Projects

Let's take a closer look at how an IDE can benefit developers through an example involving a visualization project.

### 1. Project Development with Integrated Tools

- **Example:** Min is working on a visualization project utilizing a data modeling language.
- **IDE Integration:** Min uses an IDE integrated into her enterprise-grade visualization tool, which enhances her workflow and saves development time.

### 2. Code Editing Features

- **Code Entry:** Min writes code directly into a code editor within the IDE.
- **Text Editing and Formatting:** The IDE's text editor is designed for specific programming languages, offering editing and formatting options similar to those in a word processor.
- **Autocomplete Functionality:** When Min writes in LookML (a data modeling language), the IDE's autocomplete feature predicts the code she may want to use, making her coding process faster and more efficient.
- **Syntax Highlighting:** The built-in syntax highlighter allows Min to check and correct her syntax quickly, ensuring accuracy in her code.

### 3. Debugging Capabilities

- **Error Tracking:** The IDE includes a built-in debugger that highlights potential errors and suggests fixes, enabling Min to locate and rectify issues more efficiently.
- **Improved Code Quality:** This feature helps Min produce cleaner, more functional code.

### 4. Accessing Metadata

- **Integrated Visualization Tool:** Min benefits from having access to the model's metadata directly within her development environment. This integration provides insights into the model's structure, facilitating efficient code writing.

### 5. Project Organization

- **Centralized Access:** The IDE allows Min to access all project files in one place, helping her stay organized and save time on file management.

## 6. Version Control Tools

- **Tracking Code Changes:** The IDE includes version control features that enable Min to track changes and deploy her code easily.
- **Collaboration:** Version control also enhances collaboration with other team members, allowing for seamless code sharing and feedback.

## Conclusion

While the specific features of an IDE may vary based on the tool used, integrating data tools within an IDE offers a powerful solution for developers. It not only streamlines the coding process but also provides all the necessary resources in one convenient location. By leveraging an IDE, developers can enhance their productivity and improve the overall quality of their projects.

# ▼ The Benefits of Using Version Control

## Introduction

Using a data visualization user interface (UI) can be a quick and convenient way to make changes directly to a deployed dashboard. However, it can also lead to challenges such as tracking bugs introduced by those changes or overwriting each other's work when multiple developers are involved. To address these issues, development teams often adopt a process known as **version control**.

## What is Version Control?

Version control is a system that tracks changes to code, data, or other files over time, making it easier to manage complex projects. Let's break down the key components of a version control system:

### 1. Repository:

- A repository serves as the central location for storing and managing files and the history of a project.
- Developers typically create a repository as the first step in using version control. This can be set up locally or hosted on a remote server, establishing a **single source of truth** for the project.

### 2. Branches:

- In version control, developers make changes to the project in **branches**.

- A branch is a working copy of a repository that splits off from the main project code, allowing for independent development paths.
- When a new branch is created, it duplicates the main project code, enabling developers to make changes without affecting the main project or other branches.

### 3. Independent Development:

- Since branches operate independently, developers can work on different features simultaneously without interfering with one another. This is particularly beneficial in large-scale projects, allowing for collaboration while keeping changes local to the branch.

## Collaboration in Version Control

Let's look at an example to illustrate how version control facilitates collaboration:

- **Team Collaboration:**

- Min and Carl are working on a complex dashboard, each assigned to different features.
- They start by creating their own branches in the shared repository, providing them with separate copies of the code to work on independently.

- **Seeking Help:**

- If Carl encounters a challenge, he can share his branch with Min, who can then review Carl's code, provide feedback, and help him move forward.

- **Quality Assurance:**

- Min and Carl can also share their branches with other team members for testing, helping to identify potential bugs or errors before the code goes live.

## Merging Changes

Once Min and Carl complete and test their work in their branches, they can merge their changes back into the main project code and deploy it to the live project. This process ensures that:

- **Independent Work:** Developers can focus on their assigned tasks without disrupting others.
- **Collaboration:** Team members can easily provide feedback and support each other.
- **Quality Control:** Testing and code review happen before deployment, enhancing the overall quality of the project.

## Conclusion

Managing complex visualization projects, especially in a team setting, can be challenging. Version control offers significant benefits, allowing teams to track changes over time, collaborate effectively, and maintain organized projects. By implementing version control, teams can streamline their workflow, improve communication, and ensure the successful deployment of their dashboards.

## ▼ Introduction to Data Modeling Languages

### What is a Data Model?

- A **data model** is a conceptual framework for organizing data elements and illustrating how they relate to one another.
- It can be visually represented to showcase the structure and relationships within the data.

### What is a Data Modeling Language?

- A **data modeling language** is a tool used to create and represent semantic data models.
- A **semantic data model** employs everyday language to convey the meaning of data, making it more accessible and easier to understand.

### Key Features of Data Modeling Languages

Data modeling languages come with several common features that make them suitable for building data models:

#### 1. Abstraction:

- **Abstraction** helps simplify complex concepts by focusing on their essential parts.
- Data modeling can be intricate, but using a data modeling language reduces this complexity.
- Developers can concentrate on creating accurate, efficient, and reusable models without getting bogged down in the details of implementation.
- **Example:** When using the data modeling language **LookML** to create a dimension, you define the name, type, and SQL statement needed, while the underlying LookML engine takes care of the implementation details. This user-friendly approach allows end users to interact with the data confidently.

#### 2. Modularity:

- **Modularity** refers to breaking down a system into smaller, self-sufficient parts that can be easily reused.
- This feature is particularly beneficial for developers working on large projects, especially in team settings.
- For instance, in LookML, you can create a measure once and reuse it multiple times in the same or different projects. This not only saves time but also ensures that all users rely on a consistent source of truth, as the measure behaves the same way each time it's used.

### 3. Efficiency:

- Data modeling languages enhance workflow efficiency by allowing the reuse of measures and data models, streamlining the development process.
- Many languages come with built-in validators that check code syntax and catch errors before deployment.
- Additionally, data modeling languages facilitate the generation of documentation for models, helping communicate essential information to team members who may need to reuse components you've built.

## Choosing a Data Modeling Language

As a developer, the choice of data modeling language depends on the specific task and tools you are using. Regardless of the language selected, data modeling languages are valuable tools that can significantly improve data quality and the effectiveness of your models.

## Conclusion

Data modeling languages play a crucial role in helping analysts and developers create semantic data models that are understandable, reusable, and efficient. By leveraging the features of abstraction, modularity, and efficiency, you can enhance your data modeling practices and build more effective data systems. Thank you for joining me, and I hope this overview helps you understand the importance of data modeling languages!

## ▼ Data Modeling Languages for Business Needs

### The Role of Data Modeling Languages

Data modeling languages can significantly enhance data quality and boost efficiency. However, to be truly beneficial for an organization, these languages must align with and support business needs. A data modeling language serves as a

tool for creating and representing semantic data models, which help analysts work effectively with enterprise data.

## Three Primary Advantages of Data Modeling Languages

Data modeling languages offer three key advantages that can help address business challenges:

### 1. Defining Fields in a User-Friendly Semantic Layer:

- A **semantic layer** is a framework that standardizes definitions and logic, ensuring everyone in the organization interprets the data consistently.
- For instance, a data team at a large nonprofit can utilize a data modeling language to define dimensions and measures for their donor data. This common vocabulary helps prevent misunderstandings and errors while facilitating communication across teams.
- By establishing clear definitions, everyone who interacts with the data can share a consistent understanding.

### 2. Creating a Single Source of Truth:

- Data modeling languages enable the development of a **single source of truth**, allowing users across the organization to access the same data concurrently.
- For example, a hospital's data team can employ a data modeling language to consolidate patient data into a comprehensive data model.
- With this model, they can create interactive dashboards that provide doctors and nurses with access to vital patient information. This shared access enables healthcare professionals to make timely, informed decisions that ultimately improve patient outcomes.

### 3. Building Purpose-Specific Visualizations:

- Data modeling languages facilitate the creation of visualizations tailored to specific business needs, fostering data-driven decision-making.
- Consider a financial services team that needs to monitor investment performance in near-real-time. The data team can use a data modeling language to design the structure and layout of a dashboard, incorporating interactive features and visualizations that help financial services professionals swiftly find the information they require.
- This tailored dashboard allows the financial services team to access and engage with the necessary information quickly, optimizing returns and safeguarding client investments.

## Conclusion

Whether you're part of a nonprofit, hospital, or financial services organization, as a cloud data analyst, your role will involve leveraging data to meet business needs and enhancing team collaboration with data. Data modeling languages are invaluable for achieving these objectives, as they improve communication and understanding of data, ensuring that everyone is aligned and on the same page.

Thank you for watching! I hope this overview highlights the importance of data modeling languages in addressing business needs effectively.

## ▼ Discover Dashboards as Code

### Why Dashboards-as-Code?

Data visualization tools have made it incredibly simple for even non-technical users to create dashboards and visualizations. However, as the number of dashboards within an organization expands, tracking and managing them can become challenging. That's where the dashboards-as-code approach comes in.

### What is Dashboards as Code?

Dashboards as code involves managing dashboards by defining them in code. This method offers several benefits:

1. **Change Tracking:** It allows for easy tracking of changes in dashboards.
2. **Testing Features:** Developers can test features before making them live.
3. **Reusability:** Entire dashboards can be reused repeatedly.

### Benefits of Using Dashboards as Code

Why should teams consider adopting the dashboards-as-code approach? Here are a few key reasons:

- **Trustworthy Visualizations:** Like a software product, visualizations are thoroughly reviewed, iterated upon, and tested before going live. This instills confidence in users, as they know the data has been validated.
- **Reverting Changes:** For instance, when a user modifies a dashboard using a visualization tool's UI, those changes go live immediately. If an error is introduced, reverting to a previous version can be difficult. In contrast, with dashboards as code, developers can roll back to prior versions easily if a problem arises.
- **Peer Reviews:** Changes can be peer-reviewed, minimizing the chances of errors.

- **Cross-Tool Reusability:** Because dashboards are stored as code, developers can easily reuse them across different tools by importing the code.
- **Thorough Testing:** Developers can test and validate features or entire dashboards to ensure they function as intended before being made public. This guarantees that only high-quality dashboards reach the end users.

## Challenges to Consider

While the dashboards-as-code approach has its advantages, there are also potential challenges:

1. **Learning Curve:** If you're not familiar with coding, the approach can have a steep learning curve.
2. **Tool Compatibility:** Not all visualization tools support dashboards as code. You may need to find alternative tools or learn a different coding language.
3. **Time-Consuming:** Building entire dashboards using code can be time-consuming, particularly for those new to programming.

## Conclusion

Dashboards as code is a powerful approach that can help organizations manage their dashboards effectively. However, before deciding to adopt this method, it's crucial to weigh the pros and cons to determine if it aligns with your organization's data needs. Thank you for joining me in this exploration of dashboards as code!

## ▼ Derived Tables for Complex Data Problems

### Introduction to Derived Tables

As a cloud data analyst, you typically write queries to request specific data sets from existing tables in a database. However, sometimes these queries can become complex, especially when the data you need isn't readily available in the existing tables. That's where **derived tables** come into play.

### What is a Derived Table?

A derived table is essentially a query whose results are used as if it were an actual table in the database. However, derived tables are not independent queries; they are nested within an outer query. This means they cannot be executed on their own. Instead, the outer query uses the results of the derived table to create its own results. The derived table gathers specific data needed to answer the question posed by the outer query.

Once the outer query finishes executing, the virtual table (derived table) is typically discarded and not stored in the database.

## Example: Liz the Data Analyst

Let's explore how derived tables work through an example involving Liz, a data analyst at a large non-profit organization.

1. **Understanding Donor Behavior:** Liz is tasked with identifying two types of donors:
  - **Megadonors:** Donors who contributed more than \$10,000 total in the past year.
  - **Frequent Donors:** Donors who donated more than three separate times during the same period, regardless of the donation amount.
2. **Data Challenge:** The exact data Liz needs doesn't exist in any single table in the donors' database. To address this, Liz decides to use a derived table.
3. **Creating the Derived Table:**
  - Liz identifies the data required for the outer query to run successfully.
  - She writes a query for the derived table that groups donations by donor ID and aggregates the data to count the number of donations and the total amount donated for each donor.
4. **Outer Query:**
  - Liz then writes an outer query that utilizes the results from the derived table.
  - When the outer query is executed, the nested query runs first, gathering the necessary information into a derived table.
  - The outer query then uses this information, just as it would with a regular, stored table, to identify both megadonors and frequent donors over the last year.

## Limitations of Derived Tables

While derived tables are powerful, they do have limitations:

- **Performance Impact:** Derived tables are created from scratch each time the outer query is run, which can impact performance, especially if the derived table is complex.
- **Non-Persistent:** Derived tables are not stored in the database and are not reusable in other queries. If you want to use the results of a derived table in another query, you need to recreate it as part of that query.

## Benefits of Derived Tables

Despite their limitations, derived tables offer significant benefits:

1. **Simplification of Complex Queries:** They break down complex queries into modular parts, making them easier to write, read, and execute.
2. **Integration with Visualization Tools:** Derived tables can be used with enterprise-grade visualization tools, allowing users to perform complex calculations on data and create custom views and reports.

## Conclusion

In summary, derived tables provide a powerful way to work with data and can help cloud data analysts solve complex data problems. They ensure that queries are simple yet effective. However, it's important to use derived tables wisely, weighing the trade-offs between performance and meeting business needs with your data. Thank you for joining me in this exploration of derived tables!

## ▼ Improve Performance with Caching

When it comes to data visualizations, speed and accessibility are crucial. Slow load times can negatively impact users' ability to make timely decisions with data and meet their business goals. To ensure people have access to data when they need it, cloud data analysts leverage **caching**.

### What is Caching?

Caching is the process of storing data in a temporary location so it can be accessed more quickly in the future. As a cloud data professional, you can use caching to keep frequently-used data in memory, improving the performance of your visualizations.

### Benefits of Caching

Caching offers three primary benefits that enhance visualization performance:

#### 1. Reduce Traffic to the Data Source:

- Caching stores a query's results in memory for a specified time, significantly reducing the number of queries sent to the data source.
- For instance, consider a large warehouse data team that uses a dashboard to monitor time-sensitive inventory data. Without caching, every interaction with the dashboard generates a new query to the data source, leading to excessive requests—especially during peak usage times.
- By implementing a **caching policy**, which is a set of rules determining how long cached results are stored and when they are refreshed, the data team can manage how often the database refreshes the dashboard data.

### **How It Works:**

- Before issuing a new query to the database, the system checks if the results are already cached in memory.
- If a cached copy is available, it is used for the visualization, preventing an unnecessary query to the database. If not, the new query is sent, and the results are stored in memory for future access.

### **2. Minimize Load Times:**

- Caching helps minimize the load time of visualizations because data can be retrieved from memory rather than querying the data source each time.
- This is particularly beneficial when working with large datasets, as accessing data from memory is significantly faster than querying the database.

### **3. Maintain Data Availability:**

- Caching also helps ensure the availability of data visualizations. Storing results in memory can be critical in situations where the data source is unavailable or when internet connectivity is unstable.

## **Considerations for Caching**

While caching offers significant advantages, there are important considerations to keep in mind:

### **1. Stale Data:**

- Caching can lead to stale data if not managed properly. To prevent this, it's crucial to set a reasonable time limit for cached data. The timeframe should be long enough to allow the retrieved data to be used effectively but not so long that the data becomes outdated.

### **2. Monitoring Caching:**

- Regular monitoring of caching is essential to ensure that users access up-to-date data. It's important to verify that the cache is functioning correctly—storing data when it is fresh and deleting it when it expires.

## **Conclusion**

In summary, caching can significantly enhance the performance of your visualizations, reduce load times on the database, and ensure that data is consistently available to users. However, not all caching policies are created equal. When considering a caching policy, it's important to ensure that it aligns with your organization's data needs and effectively balances performance with data accuracy.

## ▼ Pre Intermediate

### ▼ 1- Introduction to Data Analytics on Google Cloud

#### ▼ Understand the Data Analytics Lifecycle on Google Cloud

##### ▼ Course Introduction

Capturing, managing, and utilizing data is fundamental to redefining customer experiences and creating new value across nearly every industry. Data analytics serves as a critical step in this transformative journey. Organizations must be equipped to harness data, derive insights, and make data-driven decisions to activate their broader operations effectively.

In the **Introduction to Data Analytics on Google Cloud** course, you will learn about the data analytics workflow within Google Cloud and the tools available for exploring, analyzing, visualizing, and sharing data.

This course targets data analysts and anyone interested in gaining insights from their data using **BigQuery** and **Looker**.

#### Key Learning Objectives

In this course, you will learn to:

- **Describe the Data Analytics Workflow:** Understand the steps involved in data analytics.
- **Summarize Different Types of Analytics:** Get familiar with the various analytics approaches and their applications.
- **Identify Google Cloud Data Analytics Products:** Learn about the different products offered by Google Cloud for working with data and their specific uses.
- **Describe Data Sources, Structures, and Storage Options:** Understand the different data sources available in Google Cloud and how to manage them.
- **Utilize BigQuery, Looker, and Looker Studio:** Learn to answer data-related questions and influence business decisions using these powerful tools.

#### ▼ Introducing a Google Cloud Data Analytics Workflow

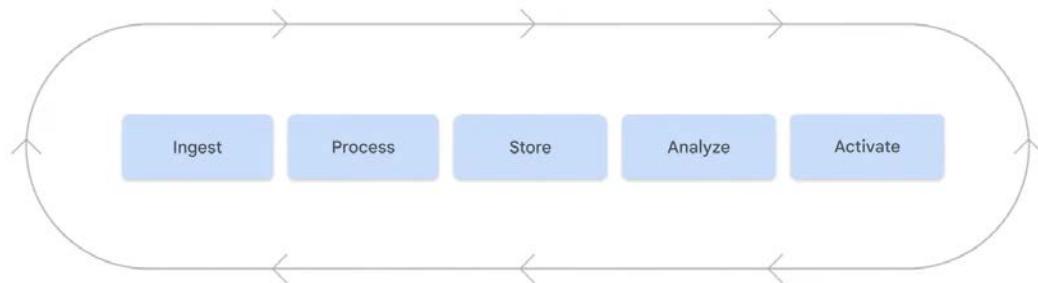
Data is an essential ingredient for driving innovation and differentiation, serving as the key to unlocking value from artificial intelligence. It comes from various sources and inputs, including operational systems, web sources, social media, and the Internet of Things (IoT). Additionally, data can be structured or unstructured,

and to generate insights, it's vital to combine and make sense of different types of data, regardless of its state or how it was created.

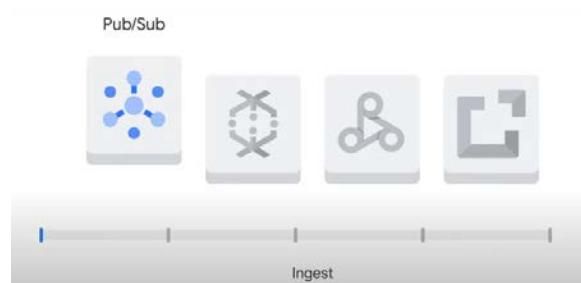
Google Cloud provides serverless, comprehensive, and integrated solutions for each step of the data analytics lifecycle. This lifecycle encompasses the processes of collecting, storing, processing, and analyzing data to extract valuable insights, making it critical for any business that aims to make informed decisions based on data.

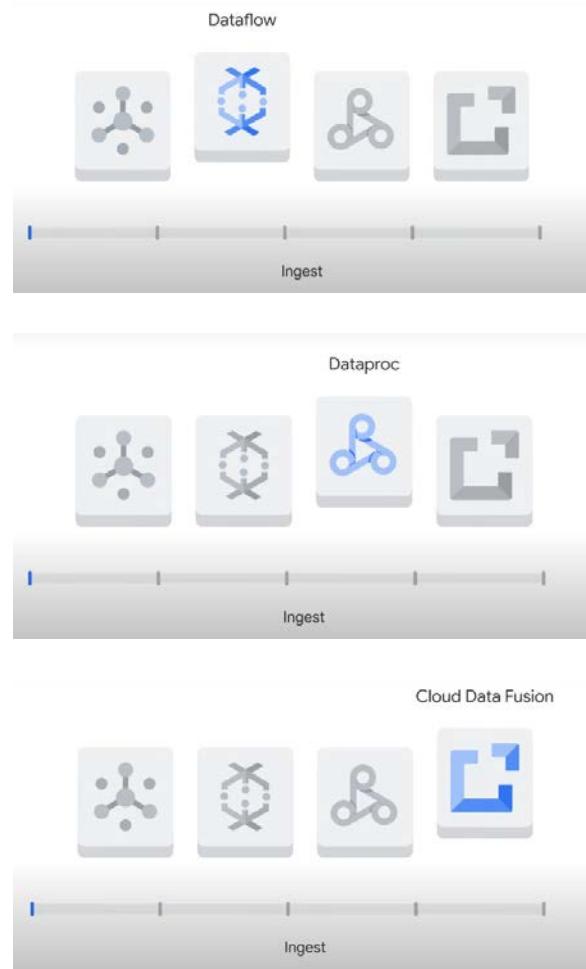
The data analytics lifecycle is an iterative process, often requiring you to revisit various steps as you learn more about your data and your objectives. Let's explore the steps involved:

## Data analytics lifecycle



1. **Ingest Data into the Cloud:** This initial step involves using Google Cloud's ingestion and processing tools to break down data silos and enhance time to insight. Key products for data ingestion and processing include **Pub/Sub**, **Dataflow**, **Dataproc**, and **Cloud Data Fusion**.





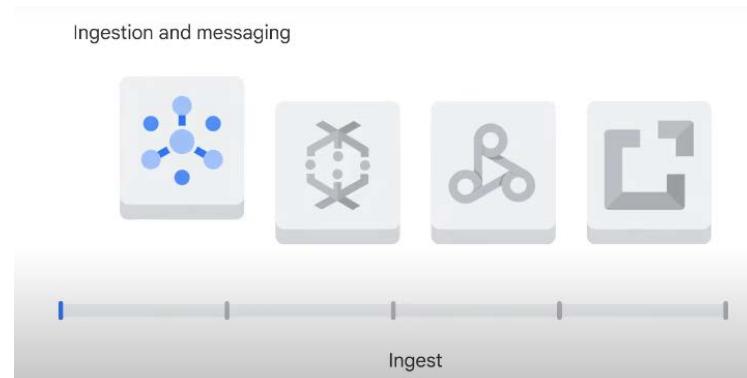
- **Cloud Data Fusion** is a fully managed, cloud-first data integration service that provides a code-free ETL (extract, transform, load) solution, integrating data across on-premises and cloud sources.



- **Pub/Sub** and **Dataflow** are streaming analytics services.

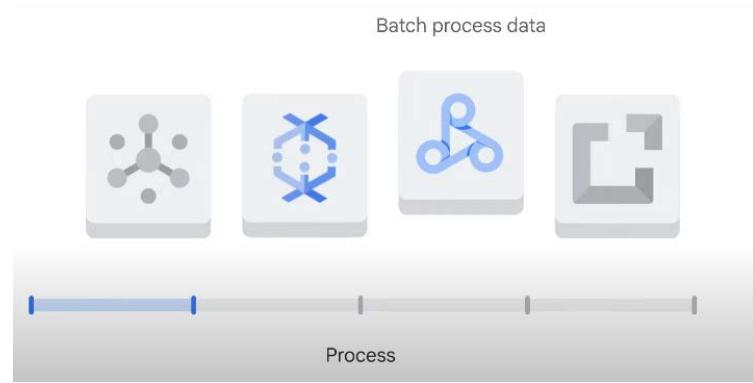


- **Pub/Sub** handles ingestion and messaging, while **Dataflow** focuses on analytics and processing.

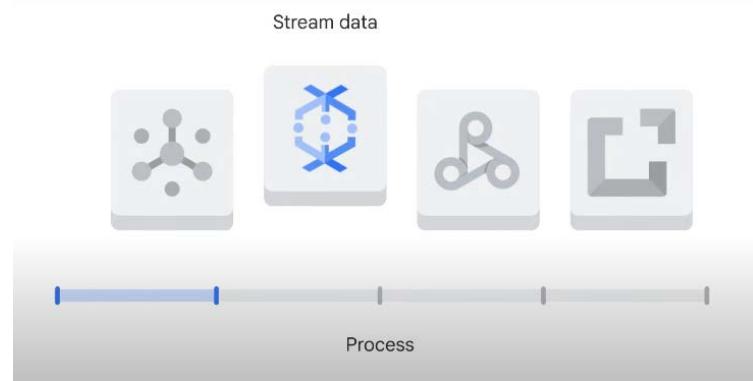


## 2. Process Data: After ingestion, data needs processing.

- You can utilize **Dataproc** for batch processing,



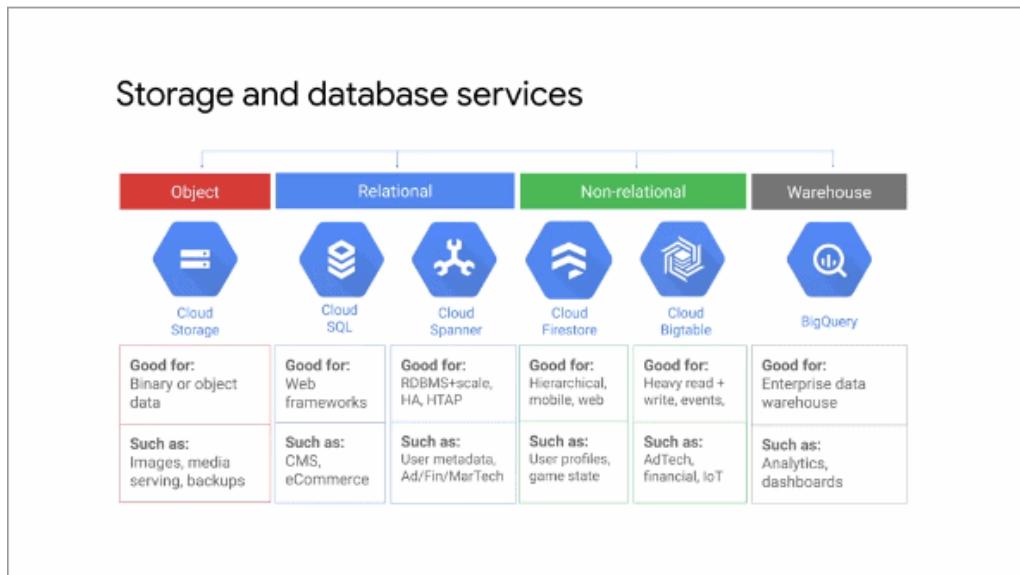
- **Dataflow** for streaming data,



- or **Cloud Data Fusion** to integrate data from multiple sources.

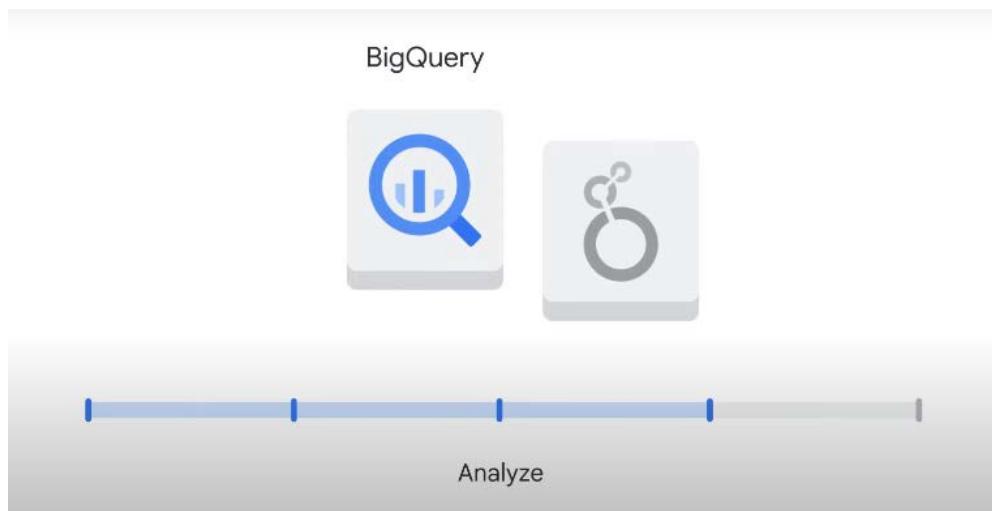


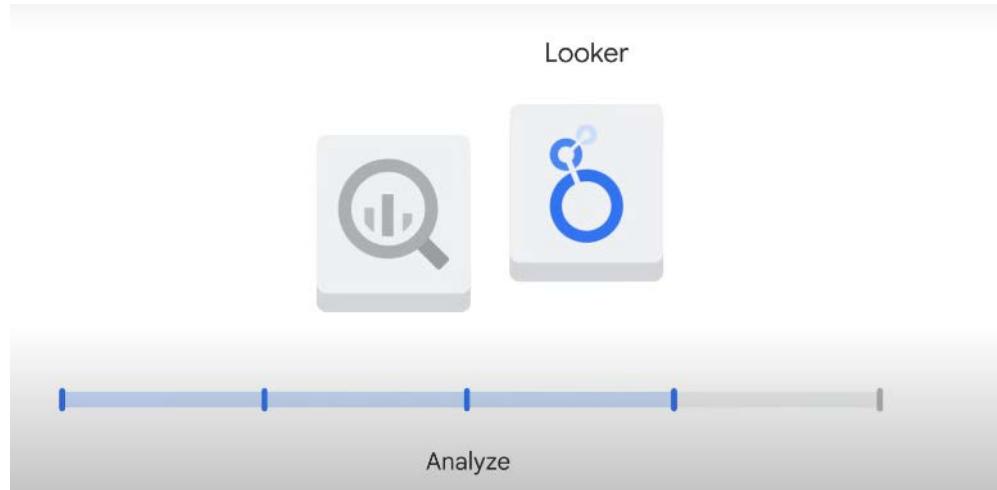
3. **Store Data Securely:** The next step involves securely storing the processed data, with solutions that scale alongside your data needs. Google Cloud offers various storage options, including:



- **Cloud Storage**
- **Cloud SQL, Cloud Spanner, AlloyDB for PostgreSQL** (relational databases)
- **Bigtable, Firestore** (NoSQL databases)
- **BigQuery** (data warehouse)

**4. Analyze Data:** Once your data is ingested, processed, and stored, it can be analyzed. This course emphasizes the analysis stage of the data analytics lifecycle, with **BigQuery** at its core. BigQuery is an elastic, flexible, secure, and reliable data warehouse that works across clouds and scales with your data. It allows for data analysis through SQL commands and is deeply integrated with Google Cloud's analytical and data processing services, enabling the construction of a cloud-first data warehouse. In addition to BigQuery, you can analyze data and visualize results using **Looker** and **Looker Studio**.





**5. Leverage Machine Learning:** Data is also key to unlocking the value of machine learning on Google Cloud. The primary product of the machine learning development platform is **Vertex AI**, which includes AutoML, Vertex AI Workbench, and TensorFlow. These products harness insights that only extensive data can provide.

Understanding the data analytics lifecycle will empower you to identify the right data, prepare it for analysis, and extract meaningful insights, positioning you for success in your data analytics journey.

## ▼ Google Cloud Data Sources and Storage Methods

Before diving into the analysis stage of data analytics, it's crucial to understand the various storage solutions offered by Google Cloud. Every application needs a method for storing data, and different applications and workloads necessitate different storage solutions. This section focuses on the storage step of the data analytics lifecycle.

| RELATIONAL  | RELATIONAL & NON-RELATIONAL   | NON-RELATIONAL (NO SQL)   | IN MEMORY   |   |   |
|---|---|---|---|---|---|
| <b>Cloud SQL</b><br>Managed MySQL, PostgreSQL, SQL Server<br>SLA: 99.95%  | <b>AlloyDB</b><br>Managed PostgreSQL-compatible, with 10x faster analytics & 4x faster transaction queries<br>SLA: 99.91%   | <b>Spanner</b><br>Multi-dialect (PostgreSQL, GoogleSQL) database, unlimited scaling<br>SLA: 99.999%   | <b>Firestore</b><br>Serverless, document database with built-in cross-client sync and offline caching<br>SLA: 99.999%   |   |   |
| Fully managed experience for low-touch administration with improved availability, security & governance, support  |   |   |   |   |   |
| <ul style="list-style-type: none"> <li>- Fastest RPS &amp; swift migrations - OLTP workloads</li> <li>- Managed experience</li> <li>- Common API &amp; control plane across database engines</li> </ul> | <ul style="list-style-type: none"> <li>- PostgreSQL-compatible that need high performance &amp; high availability or scale</li> <li>- HTAP (hybrid transactional/analytical processing)</li> <li>- Migrations off commercial databases</li> </ul> | Highest scale and availability requirements without compromising on SQL capabilities, consolidating multiple databases for cost savings                                   | <ul style="list-style-type: none"> <li>- Rapid and cost-efficient application development (no need for middle tier)</li> <li>- Easy aggregation from multiple data sources</li> </ul> |   |   |
| <ul style="list-style-type: none"> <li>- Cost-sensitive applications</li> <li>- Need high throughput &amp; consistent single-digit millisecond latencies irrespective of scale</li> </ul>               | <ul style="list-style-type: none"> <li>- Sub-millisecond latencies for reads &amp; writes</li> <li>- Cache to improve app performance &amp; throughput and reduce costs</li> </ul>  | Good For:   |   |   |   |
| Web Frameworks<br>ERP<br>CRM<br>Ecommerce & Web   | Operational Analytics<br>ERP, CRM<br>Financial Services<br>Ecommerce & Web<br>SaaS  | Order & Inventory Management<br>Online Banking, payments & ledger<br>Gaming: player profiles & gameplay data<br>Electronic Medical Records<br>Catalog Metadata Management | Web & Mobile Apps<br>News Feeds, Social Chat, Influencer Engagements<br>Game Saves, Player Profiles<br>Retail Catalogs, Point of Sales  | Real-Time Analytics (Personalization, Fraud detection)<br>IoT/Clickstream/Time Series<br>Feature Stores, Operational Data Hubs & Data Fabric<br>Batch Unstructured Data Processing<br>Financial Markets, Crypto Ledgers | Database caching<br>Session Store<br>Jobs and Queues<br>Leaderboards<br>Fast Data Ingestion |

## Core Storage Products

Google Cloud provides several core storage products, which include:

- **Cloud Storage**
- **Cloud SQL**
- **Cloud Spanner**
- **BigQuery**
- **Firebase**
- **Cloud Bigtable**
- **AlloyDB for PostgreSQL**

Depending on your specific use case, you might employ one or multiple of these services to meet your storage needs.



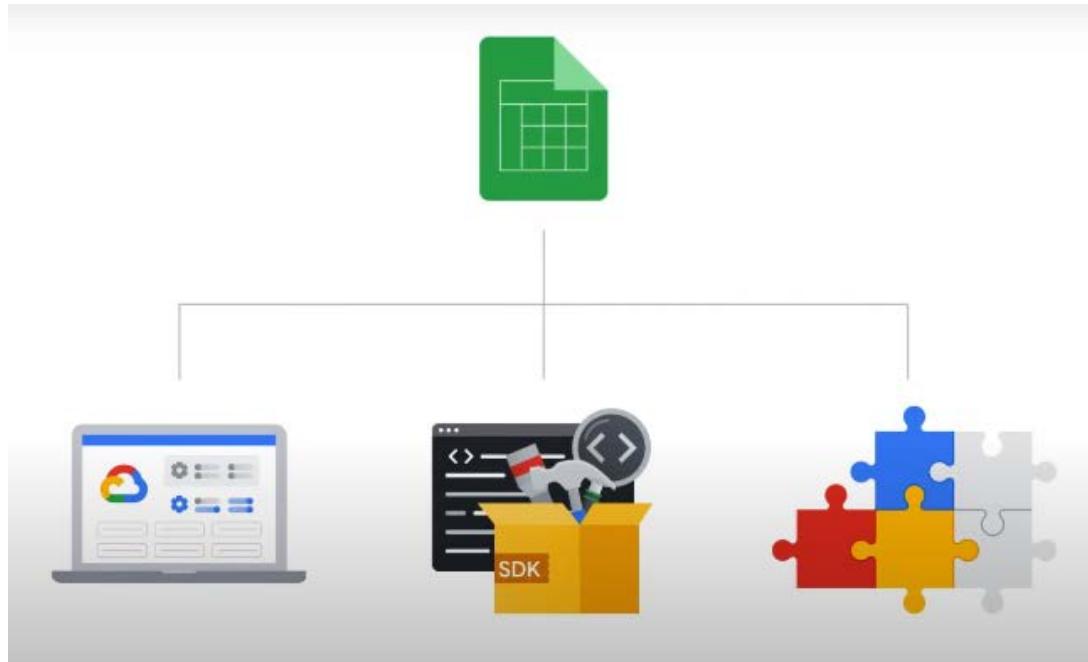
## Data Sources

Before examining the details of each storage option, it's essential to differentiate between data sources. Data sources are connectors that enable you to query data from various origins. You can use these data sources to create datasets, which are collections of data utilized for analysis and machine learning.

Google Cloud data sources can be classified into two categories:

1. **Cloud Data Sources:** These are data sources stored on Google Cloud. For example, you can connect to a Cloud Storage bucket or a Cloud SQL database.
2. **External Data Sources:** These are stored on-premises or in another cloud provider. Examples include connecting to an Amazon S3 bucket or a Microsoft SQL Server database.

You can establish a data source connection through the Google Cloud console, the Cloud SDK, or the Google Cloud API.



## Benefits of Google Cloud Data Sources

Using Google Cloud data sources provides several benefits:

- **Centralized Access:** Google Cloud data sources give centralized access to your data, simplifying the process of finding and analyzing data, regardless of its storage location.
- **Data Integration:** They facilitate the integration of data from different sources, which is valuable for building data warehouses and data lakes.

These storage options can also be leveraged to analyze your data, helping you identify trends, make predictions, and enhance business decisions. Additionally, data sources can be used for data visualization, allowing you to communicate findings effectively and make your data actionable.

## Storage Options Overview

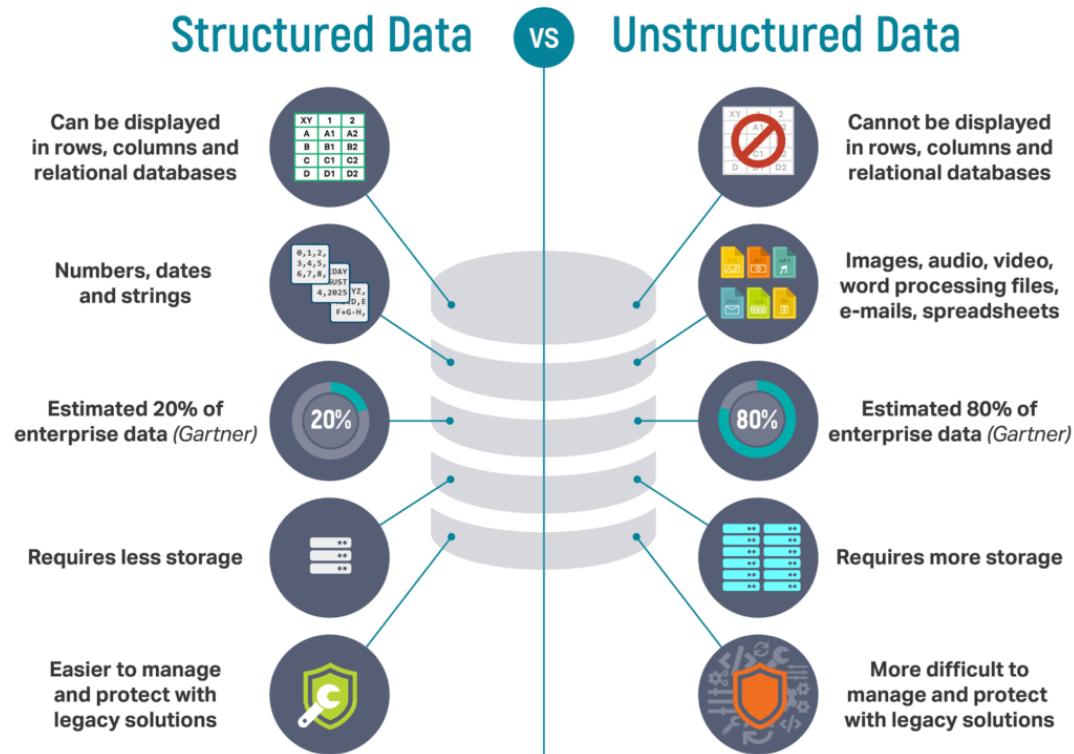
The main storage options include databases, data warehouses, and data lakes:



1. **Databases:** An organized collection of data stored in tables and accessible electronically. Google Cloud offers both relational and non-relational database options.
  - **Relational Databases:** These include Cloud SQL, Cloud Spanner, and AlloyDB for PostgreSQL, which store and manage data points related to one another. They can establish relationships between information through table joins, and Structured Query Language (SQL) is used for querying and manipulating data. Relational databases are highly consistent and reliable, making them suitable for large amounts of structured data.
  - **Non-relational Databases (NoSQL):** These databases are less structured and do not adhere to the traditional tabular format. They follow a flexible data model, making them ideal for applications with frequently changing data structures or diverse data types. **Bigtable** is a notable non-relational database product.
2. **Data Warehouses:** These are enterprise systems designed for the analysis and reporting of structured and semi-structured data from multiple sources. **BigQuery** is Google Cloud's data warehouse solution, which this course focuses on for data analysis and visualization using Looker and Looker Studio.
3. **Data Lakes:** A data lake is a repository for ingesting, storing, exploring, processing, and analyzing any type or volume of raw data, irrespective of the source. It allows for the storage of various data types in their original format, without strict pre-processing or structural requirements. Data lakes often consist of multiple products based on the nature of the ingested data.

## Complementary Roles of Data Warehouses and Data Lakes

Data warehouses and data lakes should be viewed as complementary rather than competing tools. While both store data, they are optimized for different use cases.



- **Structured Data:** This type of data is organized into rows and columns and is best suited for data analytics, making it ideal for statistical analysis and other data analytics techniques.
- **Unstructured Data:** This data does not fit neatly into tables or spreadsheets; it can include text, images, or audio. Machine learning algorithms can learn from unstructured data to identify patterns and make predictions.

Understanding these storage solutions and their respective roles will enable you to make informed decisions about how to manage your data effectively in Google Cloud.

## ▼ Summary

This concludes the section on the **Data Analytics Lifecycle**. Let's take a moment to recap what we've covered:

1. **Introduction to the Data Analytics Lifecycle:** We began by exploring the various stages of the data analytics lifecycle and identified which Google Cloud products and services are most suitable for each step.
2. **Storage Methods and Data Sources:** We discussed different storage methods, data sources, and data types relevant to analytics on Google Cloud. You learned how to choose the best storage solutions for your specific use cases.

**3. Data for Analytics vs. Machine Learning:** We also examined how data can be leveraged for analytics compared to its use in machine learning.

In the next module of this course, you will delve deeper into **finding data** and using **basic SQL commands in BigQuery** to extract valuable insights.

## ▼ Explore Data and Extract Insights by Using BigQuery

### ▼ Introduction

Welcome to this module on **exploring data and extracting insights using BigQuery**.

#### Overview of BigQuery:

- BigQuery is the central component of analytics on Google Cloud and serves as a fully-managed data warehouse.
- A data warehouse is a vast repository that stores terabytes and petabytes of data collected from various sources within an organization, aiding management decisions.

#### Module Structure:

1. **Understanding BigQuery:** You will begin by learning about BigQuery, Google Cloud's data warehouse solution.
2. **How BigQuery Works:** Next, you will discover how BigQuery operates and how to extract insights from your data effectively.
3. **Data Organization in BigQuery:** Finally, you'll learn about how BigQuery organizes data to facilitate easier analytics for data analysts.

Throughout this module, you will also view demonstrations of the BigQuery user interface.

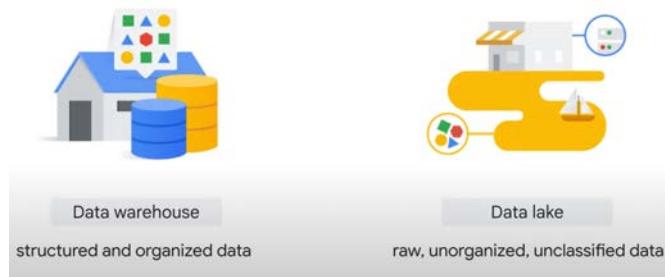
## ▼ Introduction to BigQuery for Data Analysts

### What is BigQuery?

BigQuery is a fully managed data warehouse, meaning it handles the underlying infrastructure so that users can focus on crafting SQL queries to derive business insights without concerns about deployment, scalability, or security.

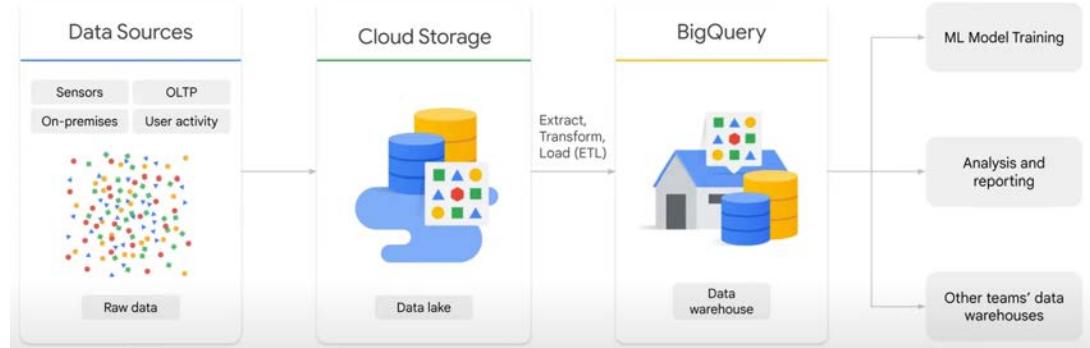
### Data Warehouse vs. Data Lake

It's essential to distinguish between a data warehouse and a data lake:

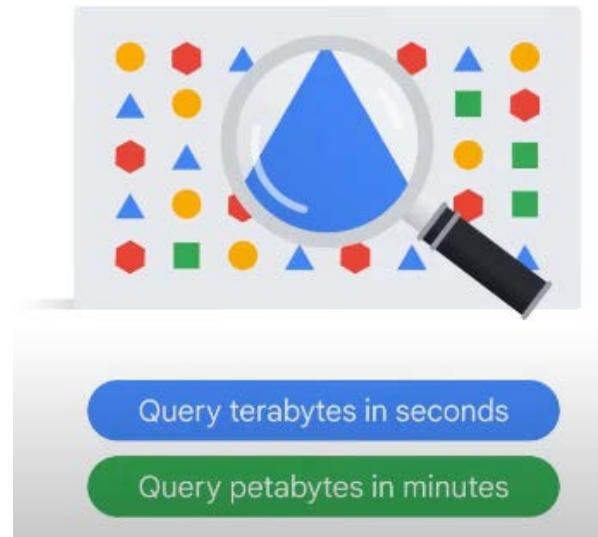


- **Data Lake:** A repository of raw, unorganized, and unclassified data without a specified purpose.
- **Data Warehouse:** A structured and organized system designed for advanced querying and analysis.

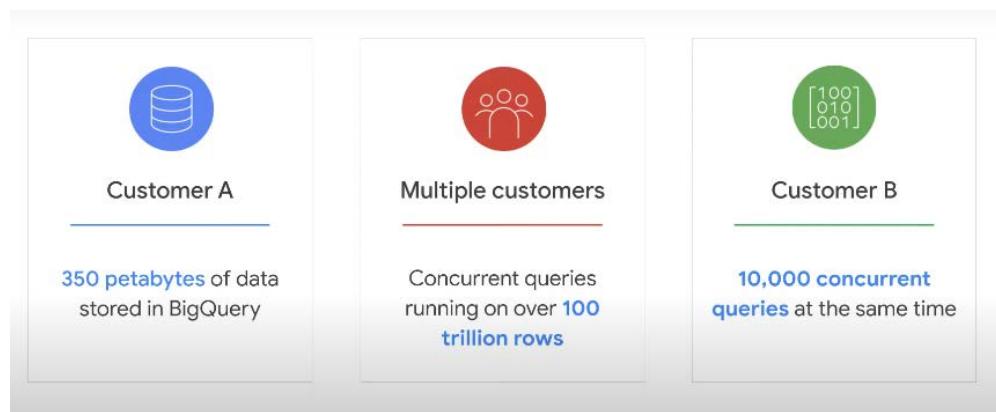
BigQuery serves as a data warehouse where transformed data can be stored for deriving business insights. Data sources flow into a data lake and are processed into the data warehouse for analysis and reporting.



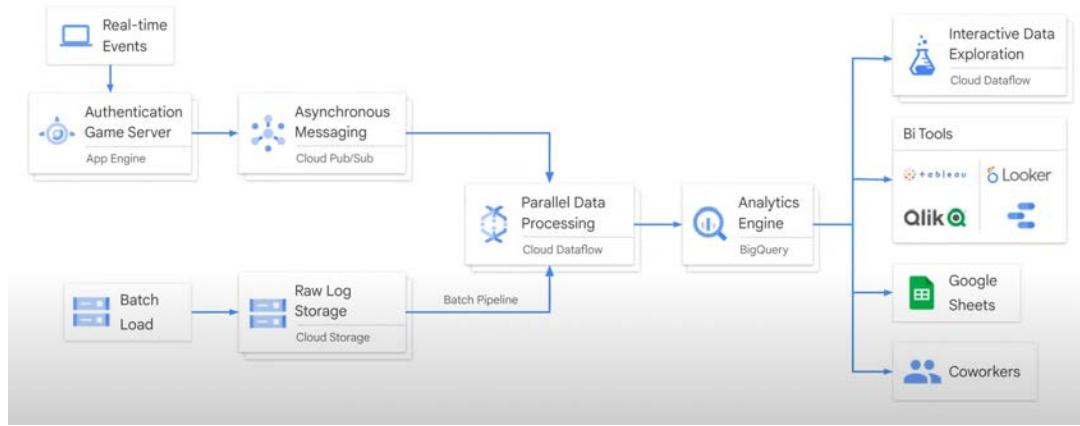
## Performance and Scalability



- BigQuery is optimized for running analytic queries on large datasets, processing terabytes of data in seconds and petabytes in minutes.
- It supports efficient analysis of massive datasets, enabling near real-time insights. For instance, one Google Cloud customer has over 350 PB of data stored in BigQuery, and others have executed queries on more than 10 trillion rows.



## BigQuery Architecture



- BigQuery functions as the analytics engine at the end of the data pipeline, storing incoming data and facilitating analysis and model building.
- It consists of two main components: a fast SQL query engine and a fully managed storage layer for datasets.



## Key Features of BigQuery



BigQuery

**Two services in one**  
Store and analyze petabytes of data.

**Serverless**  
BigQuery handles infrastructure, so you can focus on running SQL queries.

**Flexible pay-as-you-go pricing model**  
Pay for what your query processes, or use a flat-rate option.

**Data encryption at rest by default**  
Encrypt data stored on a disk.

**Built-in machine learning**  
Write ML models directly in BigQuery using SQL.

## Getting Started with BigQuery

To demonstrate the capabilities of BigQuery, we will log into the user interface and perform a query on a substantial dataset.

### User Interface Navigation

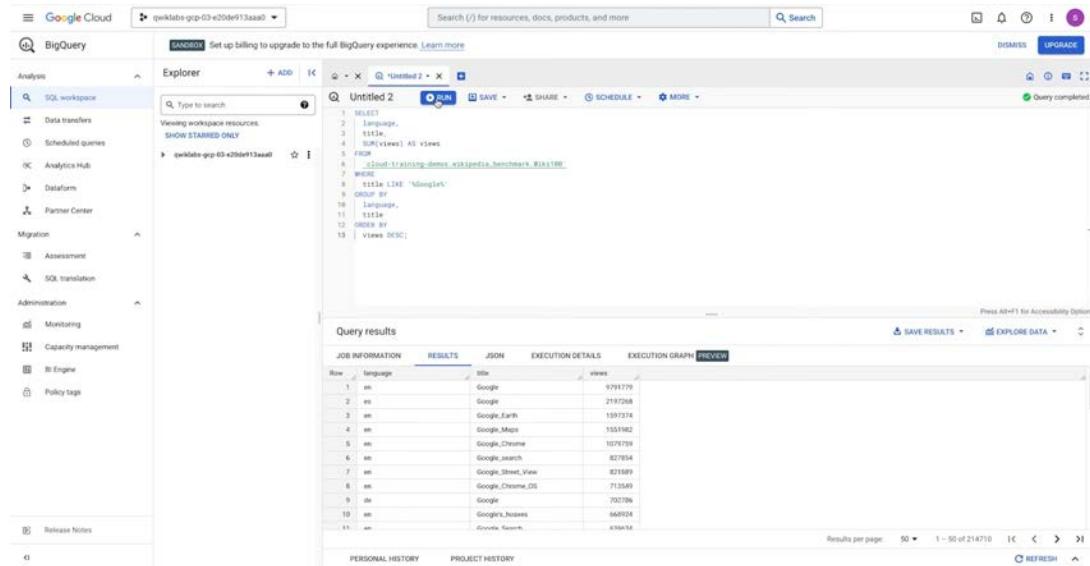
The screenshot shows the Google Cloud BigQuery interface. On the left, there's a navigation sidebar with sections like Analysis, Migration, Administration, and Partner Center. The main area is titled 'Welcome to your SQL Workspace!' and features a 'Try the Google Trends Demo Query' section with a 'OPEN THIS QUERY' button. Below this are sections for 'Add your own data' (Local file, Google Drive, Google Cloud Storage) and links to various guides. A search bar at the top right allows users to search for resources, docs, products, and more.

- BigQuery can be accessed through the Google Cloud navigation menu.
- Every resource exists within a project, linking usage to billing.
- BigQuery provides numerous public datasets, including Wikipedia page metadata.

### Example Query

Let's execute a SQL query to search for the term "Google" in the titles of Wikipedia pages, scanning 10 billion rows.

## SQL Query Example:

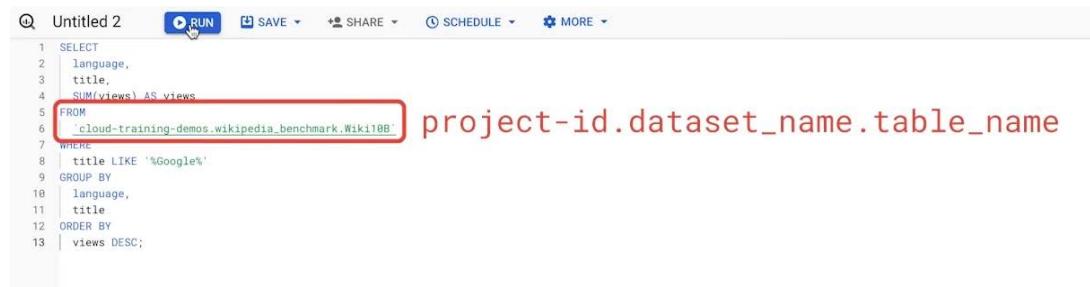


The screenshot shows the Google Cloud BigQuery interface. On the left, the sidebar includes sections for Analysis, Migration, Administration, and Release Notes. The main area is titled 'Untitled 2' and contains a SQL query. The query retrieves data from the 'cloud-training-demos.wikipedia\_benchmark.Wiki10B' dataset, filtering for titles containing 'Google', grouping by language and title, and ordering by views in descending order. The results table shows 10 rows of data, with the first few rows being: Row 1: en, Google, 9791779; Row 2: es, Google, 2197268; Row 3: en, Google\_Earth, 1597274; Row 4: en, Google\_Maps, 1551982; Row 5: en, Google\_Search, 1079739; Row 6: en, Google\_Street\_View, 821699; Row 7: en, Google\_Chrome\_OS, 713549; Row 8: de, Google, 703798; Row 9: en, Google\_Reader, 669524; Row 10: en, Google\_News, 476932.

This query showcases how BigQuery's scalable, distributed analysis engine can efficiently handle filtering, grouping, and ordering on a dataset containing billions of records.

## Understanding BigQuery Resources

- In BigQuery, data is organized within datasets, which contain tables and columns.

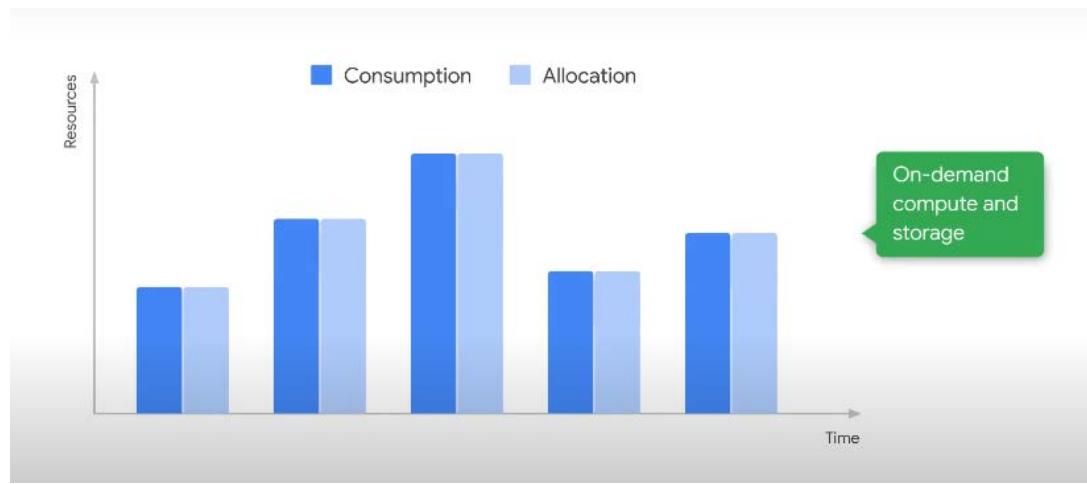


The screenshot shows the Google BigQuery query editor. A red box highlights the 'FROM' clause of the SQL query, which specifies the dataset and table name: 'project-id.dataset\_name.table\_name'. The rest of the query is as follows:

```
1 SELECT
2   language,
3   title,
4   SUM(views) AS views
5 FROM
6   cloud-training-demos.wikipedia_benchmark.Wiki10B
7 WHERE
8   title LIKE '%Google%'
9 GROUP BY
10  language,
11  title
12 ORDER BY
13  views DESC;
```

- Access to data is controlled at various levels, ensuring security.
- BigQuery Slots:** A virtual CPU unit used for executing SQL queries, with resources dynamically allocated based on usage patterns. Customers start with access to 2,000 slots for query operations.

## Query Execution and Slot System



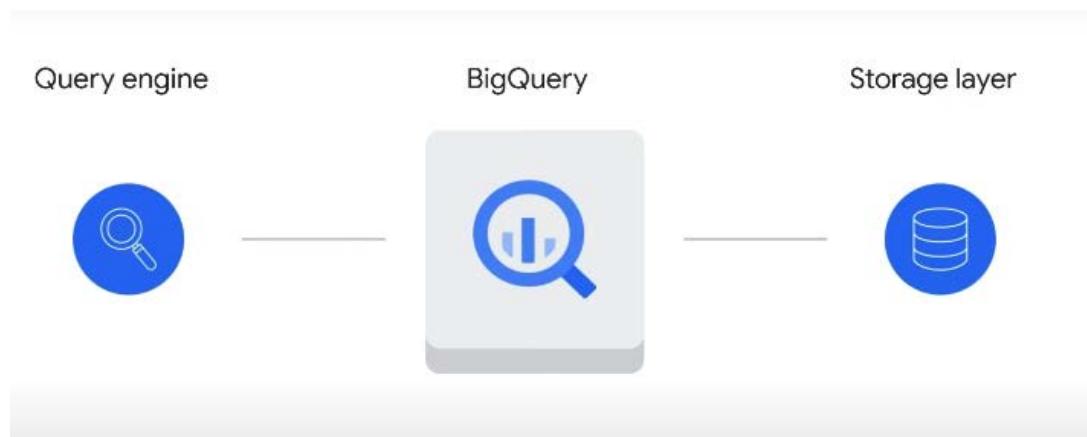
BigQuery's scalability comes from its slot system. Each query uses some number of slots, which are units of computation consisting of CPU and RAM. BigQuery dynamically allocates the necessary storage and compute resources based on the type and complexity of each query, ensuring efficient use of resources without requiring manual provisioning.

## ▼ Deriving Insights from Data with BigQuery

In this section, we will delve into the technical aspects of BigQuery, focusing on how its analytics and storage services work together to facilitate data-driven innovation. Understanding these components will enhance your ability to derive meaningful insights from your data.

## BigQuery Architecture: Analytics and Storage

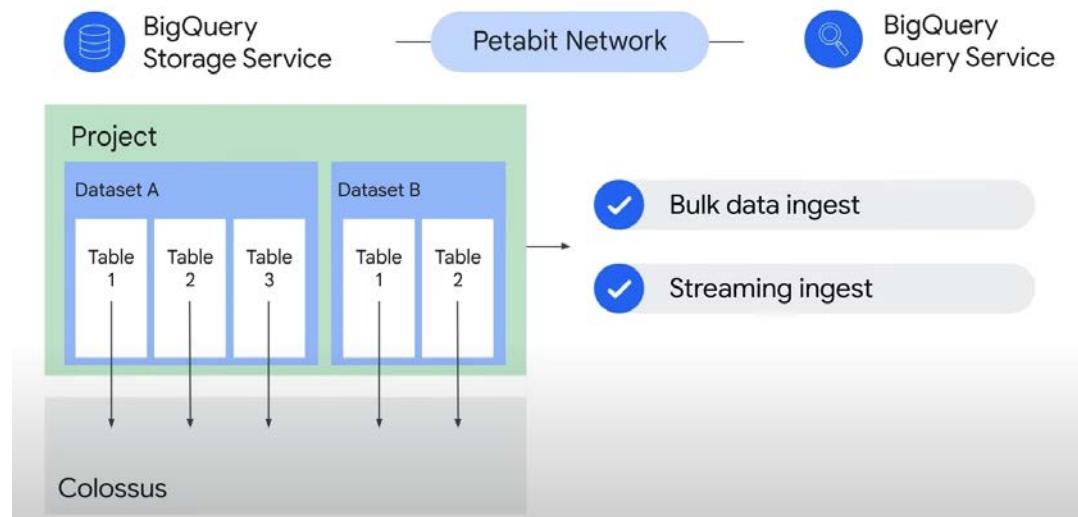
BigQuery comprises two primary services: analytics and storage. These services are interconnected via Google's high-speed internal network, which allows for the seamless separation of compute and storage resources.



## Fully-Managed Services

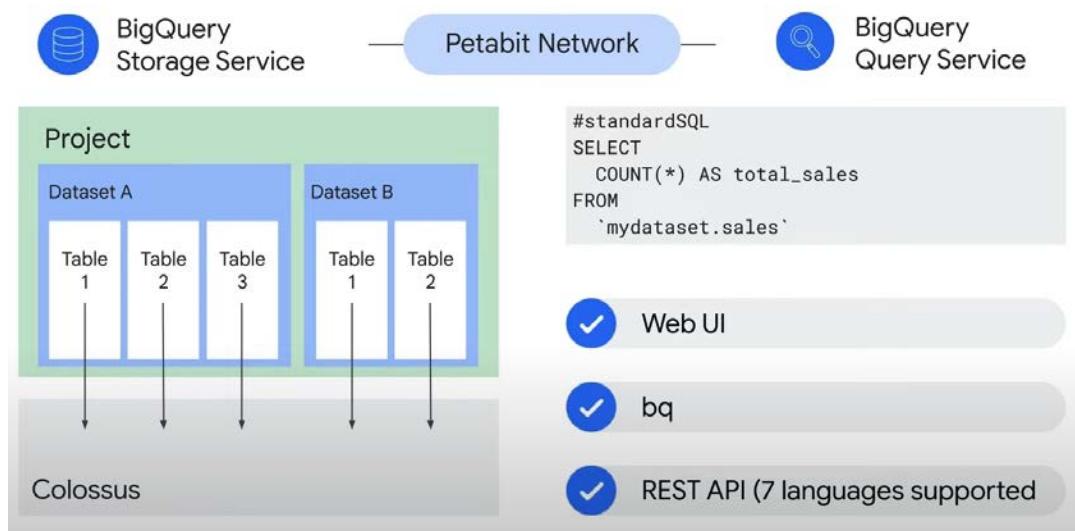
Both services are fully managed, meaning users do not need to concern themselves with how BigQuery handles data storage or automatically scales resources for large queries. However, understanding how these services operate can provide valuable insights into their performance and capabilities.

### 1. BigQuery Storage Service

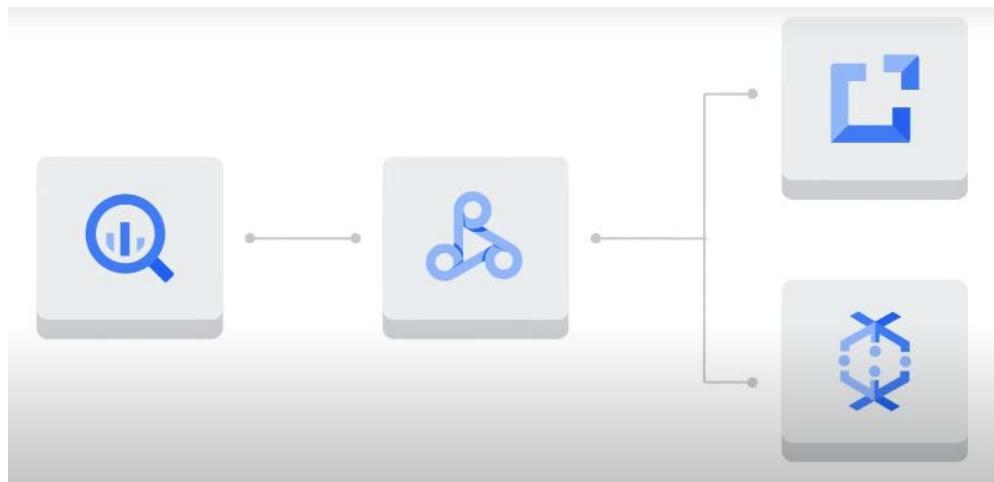


- The **storage service** automatically manages the data ingested into BigQuery.
- Data is organized into **datasets**, which are scoped to your Google Cloud project. When referencing a table in SQL queries or code, you use the format: `project.dataset.table`.
- Tables are stored as highly compressed columns in Google's **Colossus file system**, ensuring durability and availability.
- The storage service supports both **bulk data ingest** and **streaming ingest**, enabling the handling of large data volumes as well as real-time data streams.

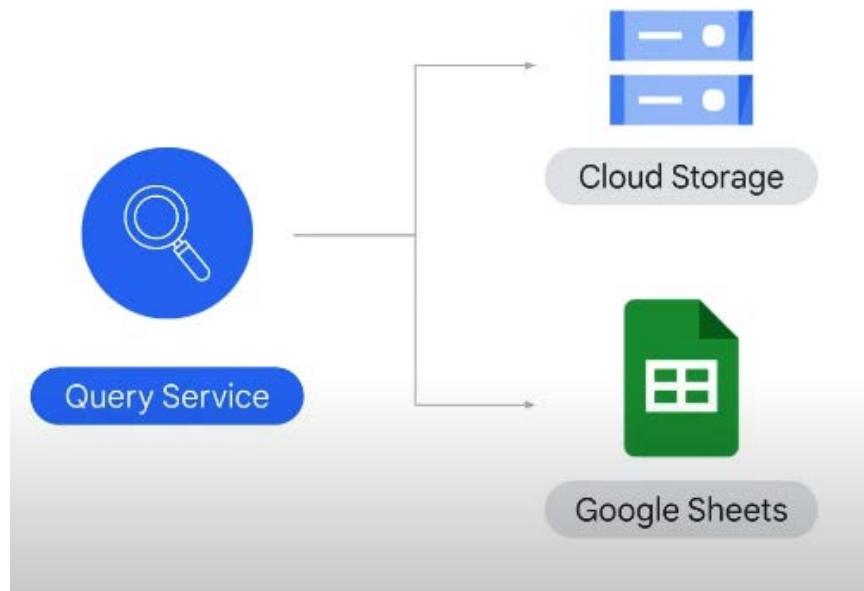
### 2. BigQuery Query Service



- The **query service** can execute interactive or batch queries submitted through various interfaces, including the console, the BigQuery web UI, the `bq` command-line tool, or the REST API, which supports seven programming languages.
- As demonstrated earlier, we utilized the BigQuery web UI for submitting our queries.
- There are connectors to other services such as Dataproc, which simplify creating complex workflows between BigQuery and other Google Cloud data processing services.



- This service can also run queries on data located in external sources, such as CSV files in **Cloud Storage** or data in **Google Sheets**. However, it's worth noting that BigQuery is most efficient when working with data stored within its own storage service.



## Integration of Services



The storage and query services work in tandem to optimize data organization and query efficiency over massive datasets, sometimes exceeding petabytes in size. They even enhance query processing jobs after submission to improve performance.

## Managing Resources and Costs

One of the key controls you have over resource consumption and costs in BigQuery is the ability to manage the amount of data processed during queries. Here are some best practices:

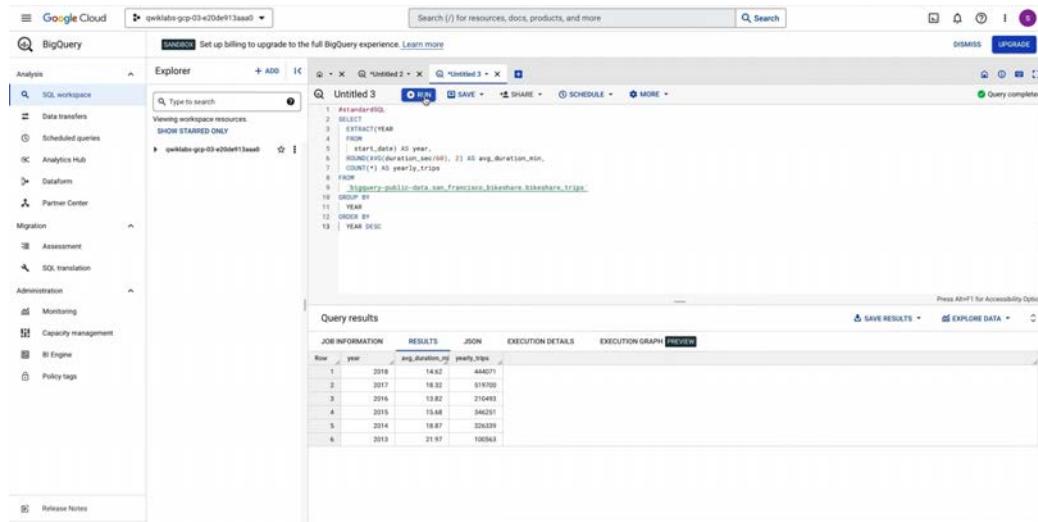
- Only select the columns you need to process and return, which minimizes data processing costs.
- Start with broad queries when exploring datasets and gradually narrow your focus to critical fields and rows necessary for your analysis.

Exploring your dataset using SQL is an essential first step in uncovering hidden insights.

## Example Query: Analyzing Bike Share Trips

Let's return to the BigQuery user interface to look at a practical example. In this case, we will run a query to count the total number of trips taken in the **San Francisco bike share** public dataset.

1. **Counting Total Trips:** The query counts all trips recorded in the dataset.



The screenshot shows the Google Cloud BigQuery interface. On the left, the sidebar includes sections for Analysis, Explorer, Data transfers, Scheduled queries, Analytics Hub, Dataform, Partner Center, Migration, Assessment, SQL translation, Administration, Monitoring, Capacity management, BI Engine, and Policy tags. A 'Release Notes' link is also present. The main area shows a query editor titled 'Untitled 3' with the following SQL code:

```
1  #standardSQL
2  SELECT
3      EXTRACT(YEAR
4          FROM
5              `start_date`) AS year,
6      ROUND(AVG(`duration_sec`), 2) AS avg_duration_min,
7      COUNT(*) AS yearly_trips
8  FROM
9      `bigquery-public-data.san_francisco_bikeshare.trips`
10 GROUP BY
11     year
12 ORDER BY
13     year DESC
```

Below the code, the 'Query results' section displays the following data:

| Row | year | avg_duration_min | yearly_trips |
|-----|------|------------------|--------------|
| 1   | 2018 | 14.62            | 444071       |
| 2   | 2017 | 18.32            | 519700       |
| 3   | 2016 | 13.82            | 216493       |
| 4   | 2015 | 19.48            | 346251       |
| 5   | 2014 | 18.87            | 326339       |
| 6   | 2013 | 21.97            | 100563       |

2. **Filtering for Data Quality:** To address concerns about data quality, we can use SQL to filter for anomalies. For instance, we can exclude customer records without a birth date:

```

# Top 5 stations for casual bike share users
SELECT
    start_station_name,
    COUNT(*) AS total_trips
FROM
    `bigquery-public-data.san_francisco_bikeshare.bike
share_trips`

WHERE c_subscription_type = 'Customer'
AND member_birth_year IS NOT NULL

GROUP BY start_station_name
ORDER BY total_trips DESC
LIMIT 5

```

- In this query, we apply the `GROUP BY` clause to group results by the **starting station name**, the `ORDER BY` clause to sort the results, and the `LIMIT` clause to display the top 5 results.

By understanding how BigQuery's storage and query services work together, you can effectively derive insights from large datasets. Through careful management of resources and a structured approach to querying data, you can uncover valuable insights that inform business decisions and drive data-driven innovation.

## ▼ BigQuery: Qwik Start - Console

[https://www.cloudskillsboost.google/parts/18/course\\_templates/578/labs/507297](https://www.cloudskillsboost.google/parts/18/course_templates/578/labs/507297)

## ▼ BigQuery Organizes Data to Make Analytics Easier

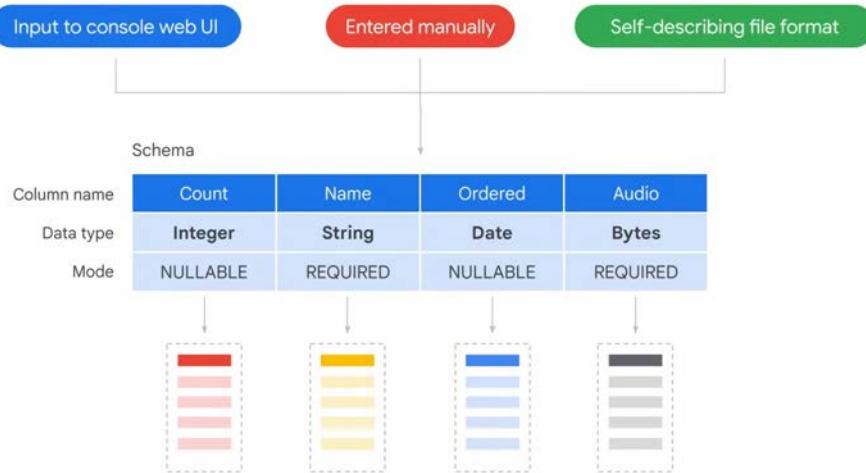
In addition to providing fast query execution times, BigQuery efficiently manages the **storage and metadata** for your datasets. Let's explore how BigQuery organizes data to facilitate exploration and analytics.

### Structuring Information in BigQuery:



- **Multiple Scopes:** BigQuery uses three main scopes—**project**, **dataset**, and **table**—to help you structure your information logically.
  - Projects can isolate datasets according to business needs, while datasets can separate tables related to different analytical domains.
- **Billing and Access Control:** You can align projects with billing, and datasets can be used for access control, ensuring that only authorized users can access specific datasets.

### Tables and Schemas:



- Every table in BigQuery has a **schema**, which defines the structure of the data.
- You can manually enter the schema through the Google Cloud Console or supply it via a JSON file.

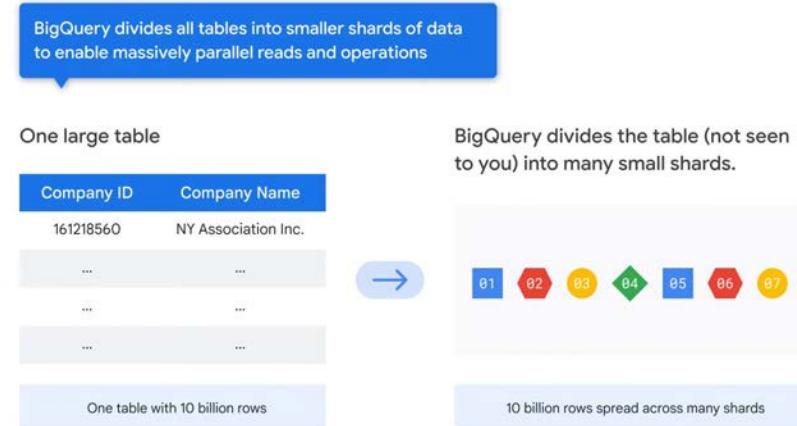
- BigQuery can also automatically infer the schema from self-describing file formats such as Avro, Parquet, ORC, Firestore export, or Datastore export.

### Columnar Storage:



- BigQuery is a **columnar store**, designed for processing columns rather than rows. This means:
  - It only reads the relevant columns to execute a query, optimizing it for the high read requirements typical of data warehouses.
  - Each column contains data of the same type, allowing for more effective compression.

### Data Management and Storage:



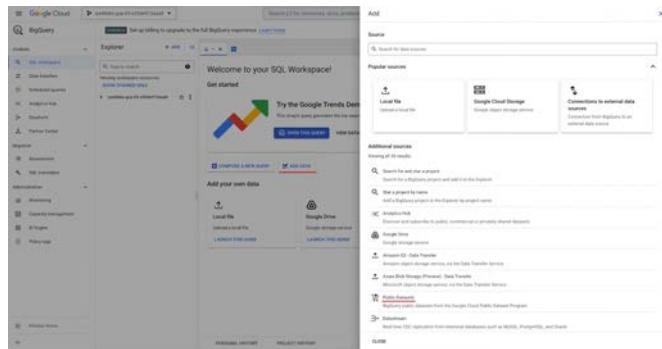
- Each column in BigQuery is automatically compressed, encrypted, and replicated to ensure availability and durability.

- BigQuery stores data in a massively distributed format, similar to Google Cloud Storage, which supports applications like Gmail and Ads.
- The backend auto-optimizes the balancing and partitioning of data shards.
- Alternatively, you can also partition and cluster your tables based on your access patterns to control costs and make your queries more performant.

## Demo: Analyzing Data with SQL in BigQuery

Now, let's demonstrate how to use SQL in BigQuery to analyze data. After this demonstration, you will have hands-on experience in a lab session.

### 1. Adding Data:



- Click **Add Data** and select **Public Datasets**.
- Search for the **NYC bike** dataset and click **View Dataset**.

### 2. Exploring the Dataset:

- In the BigQuery console, you'll see two projects in the left pane: your **Qwiklabs project ID** and **bigrquery-public-data**.

- Scroll down to find the relevant table, select it, and click on the **Preview** tab to examine its columns and data.

### 3. Running Queries:

- Start by running a query to determine the **typical duration for the 10 most common one-way rentals**.

The screenshot shows the Google BigQuery web interface. At the top, there are tabs for 'Untitled' and 'Untitled 2'. The 'Untitled 2' tab is active, displaying a SQL query:

```

1 SELECT
2   MIN(start_station_name) AS start_station_name,
3   MIN(end_station_name) AS end_station_name,
4   APPROX_QUANTILES(tripduration, 10)[OFFSET (5)] AS typical_duration,
5   COUNT(tripduration) AS num_trips
6 FROM
7   `bigquery-public-data.new_york_citibike.citibike_trips`
8 WHERE
9   start_station_id != end_station_id
10 GROUP BY
11   start_station_id,
12   end_station_id
13 ORDER BY
14   num_trips DESC
15 LIMIT
16   10

```

Below the query, the 'RUN' button is highlighted. To the right of the query area are buttons for 'SAVE', 'SHARE', 'SCHEDULE', and 'MORE'. The 'RESULTS' tab is selected in the navigation bar below the query editor.

The 'Query results' section displays the following data:

| Row | start_station_name               | end_station_name            | typical_duration | num_trips |
|-----|----------------------------------|-----------------------------|------------------|-----------|
| 1   | 12 Ave & W 40 St                 | West St & Chambers St       | 1336             | 18667     |
| 2   | W 21 St & 6 Ave                  | 9 Ave & W 22 St             | 264              | 17509     |
| 3   | E 42 St & Vanderbilt Ave         | W 33 St & 7 Ave             | 480              | 16228     |
| 4   | W 21 St & 6 Ave                  | W 22 St & 10 Ave            | 348              | 15120     |
| 5   | West St & Chambers St            | 12 Ave & W 40 St            | 1322             | 14353     |
| 6   | E 42 St & Vanderbilt Ave         | W 41 St & 8 Ave             | 437              | 14171     |
| 7   | E 42 St & Vanderbilt Ave         | Broadway & W 32 St          | 415              | 13516     |
| 8   | E 42 St & Vanderbilt Ave         | E 24 St & Park Ave S        | 398              | 13287     |
| 9   | Centre St & Chambers St          | Cadman Plaza E & Tillary St | 1506             | 12280     |
| 10  | Grand Army Plaza & Central Pa... | Broadway & W 60 St          | 606              | 12204     |

- Next, run another query to find the **total distance traveled by each bicycle** in the dataset.

The screenshot shows the BigQuery web interface. In the top navigation bar, there are tabs for 'Untitled' (selected), 'citibike\_trips', 'Untitled 2', and '\*Untitled 3' (which is currently active). Below the tabs is a toolbar with buttons for 'RUN', 'SAVE', 'SHARE', 'SCHEDULE', and 'MORE'. The main area contains a code editor with the following SQL query:

```

1 WITH
2   trip_distance AS (
3     SELECT
4       bikeid,
5       ST_Distance(ST_GeogPoint(s.longitude,
6           s.latitude),
7           ST_GeogPoint(e.longitude,
8           e.latitude)) AS distance
9     FROM
10    `bigquery-public-data.new_york_citibike.citibike_trips`,
11    `bigquery-public-data.new_york_citibike.citibike_stations` AS s,
12    `bigquery-public-data.new_york_citibike.citibike_stations` AS e
13   WHERE
14     start_station_name = s.name
15     AND end_station_name = e.name)
16   SELECT
17     bikeid,
18     SUM(distance)/1000 AS total_distance
19   FROM
20     trip_distance
21   GROUP BY
22     bikeid
23   ORDER BY
24     total_distance DESC
25   LIMIT
26     5

```

## Query results

| JOB INFORMATION |        | RESULTS        |  | JSON | EXECUT |
|-----------------|--------|----------------|--|------|--------|
| Row             | bikeid | total_distance |  |      |        |
| 1               | 19455  | 8049.65759...  |  |      |        |
| 2               | 17955  | 7819.16960...  |  |      |        |
| 3               | 15731  | 7782.78534...  |  |      |        |
| 4               | 18104  | 7728.38978...  |  |      |        |
| 5               | 17747  | 7710.99578...  |  |      |        |

- This time, note that you are pulling data from multiple tables (e.g., `citybike_trips` and `citibike_stations`) and joining them to achieve your analysis goals.

## ▼ Summary

You've reached the end of this module on **BigQuery**, Google's fully managed enterprise data warehouse for analytics and storage. Here's a brief recap of what you learned:

### 1. Introduction to BigQuery:

- You received an overview of BigQuery, along with a demonstration of the platform's key features.

## 2. Deriving Insights:

- You learned how to extract insights from your data using BigQuery, including how to access the necessary data and run basic SQL queries.

## 3. Data Storage and Organization:

- You explored how BigQuery stores and organizes data, optimizing it for efficient analytics.

Throughout the module, you viewed demos and participated in hands-on labs, enhancing your understanding of BigQuery's capabilities.

In the next section, you will learn about **Looker**, Google Cloud's data analytics and visualization tool.

## ▼ Make Data-Driven Decisions by Using Looker

### ▼ Make Data-Driven Decisions Using Looker: Introduction

Welcome to this section on **Looker**, where you will learn the fundamentals of analyzing, visualizing, and sharing data using the Looker platform. With Looker, you can activate and utilize the data you've analyzed to empower your teams to make effective, data-driven decisions.

#### What's Looker?

Looker is a powerful tool that helps you: Access and review the data that your company collects.

- **Access and review** the data that your company collects.
- **Answer data questions** in real-time.
- **Stay up-to-date** with the status of your business.
- **Use data** to drive decision-making.

#### Common Data Analysis Process in Looker

## Data analysis



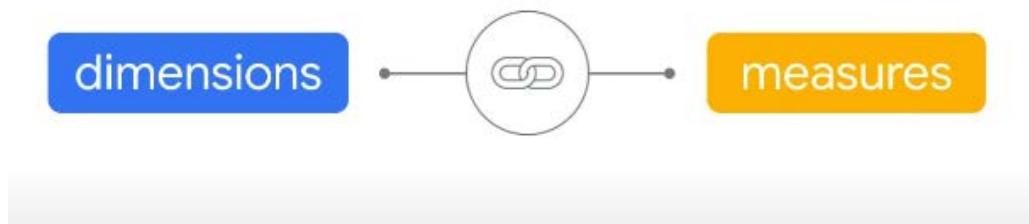
### 1. Identify Data Questions:

- Start by identifying the questions you want to answer. For example, as an analyst for an e-commerce retailer, you might ask:
  - How many sales were made last week?
  - What are your user or customer demographics?
  - What is the shipment status of all orders?

### 2. Identify Required Data:

- Determine what data you need to answer your questions. For example, to find the shipment status of all orders, you would require data on **orders**, **shipments**, and possibly **distribution center** information. If analyzing users, you might need data on traffic sources and locations.

### 3. Analyze the Data:



- In Looker, the analysis process is called "**exploring**." After identifying your questions and the necessary data, you can combine dimensions and measures in Looker to uncover answers.

### 4. Interpret the Results:



- Once you analyze and visualize the data, interpret the results to gain insights, take action, and make data-driven decisions. For instance, if you find that a large percentage of customers were directed to your website through a successful marketing campaign, you can advocate for similar future campaigns to increase sales.

## ▼ Use Looker to Analyze and Chart Data



Let's begin by exploring the **Looker interface** and detailing the main components used for data exploration.

### Understanding the Looker Interface

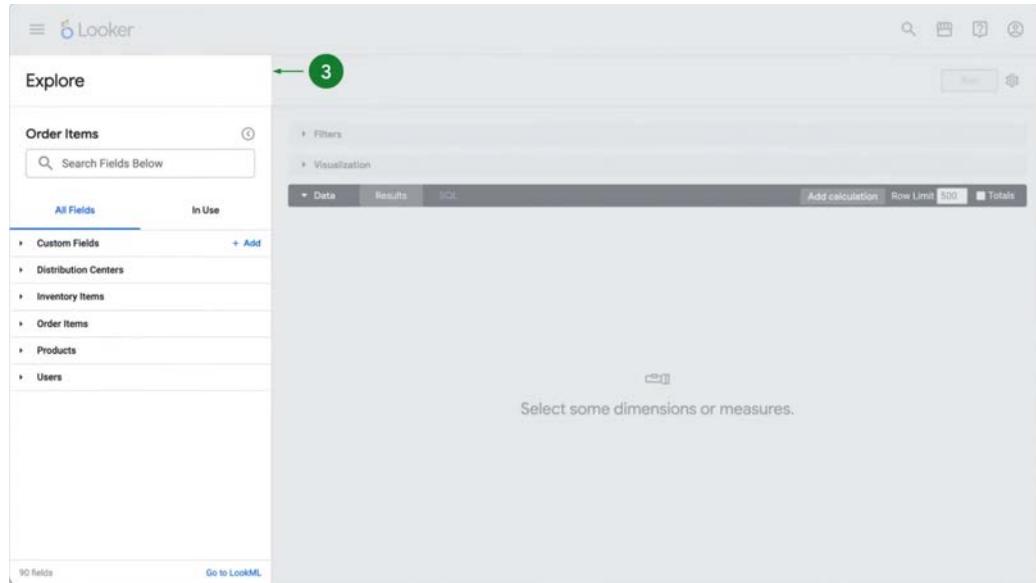
- In Looker, an **Explore** is a report-builder interface and a portal for asking questions using your data.
- Explores are curated by Looker developers using a proprietary templating language called **LookML**.
- The Explores available to you are listed in the **Explore section**.

The image consists of two vertically stacked screenshots of the Looker web application. Both screenshots show the same basic layout: a top navigation bar with search and filter icons, a left sidebar with navigation links like 'Explore' (highlighted with a green circle and arrow), 'Develop', and 'Admin', and a main content area with sections for 'Your favorite dashboards and Looks', 'Recently viewed at your organization', and 'Popular at your organization'. The main content area includes descriptive text and small visual icons.

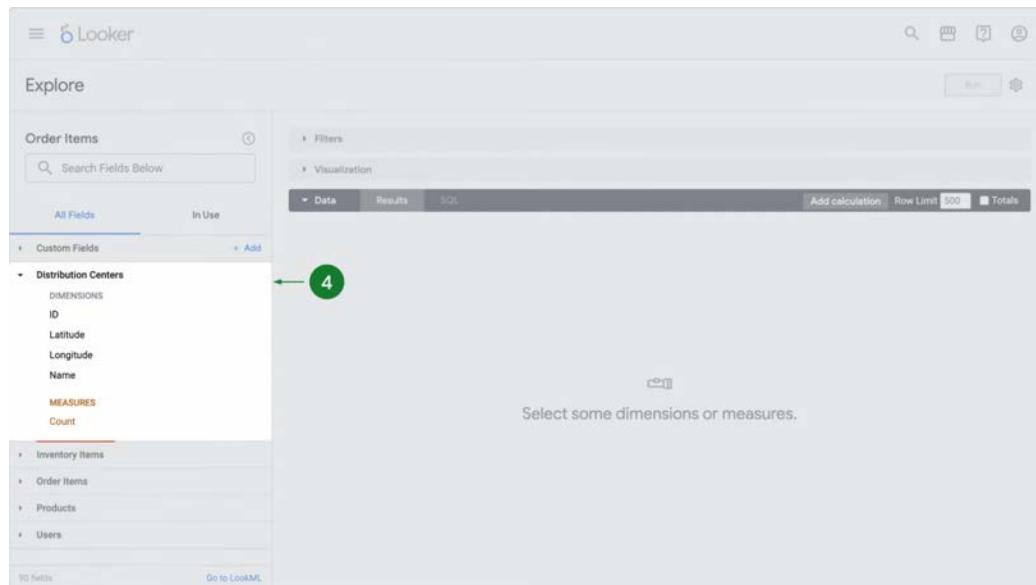
**Screenshot 1:** The 'Explore' link in the sidebar is highlighted with a green circle and a blue arrow pointing to it. This indicates the current active section of the application.

**Screenshot 2:** The 'Explore' link in the sidebar is highlighted with a green circle and a blue arrow pointing to it. This indicates the current active section of the application. Additionally, the left sidebar itself is highlighted with a gray background, showing expanded categories such as 'E-Commerce Training' (Events, Order Items), 'FAA' (Airports, Flights), and 'Looker Basics' (Fruit Basket). A small circular icon with a question mark is visible near the bottom right of the sidebar.

- This Explore example contains ecommerce data.
- The left side panel displays the **field picker**, where fields are bundled into groups called **views**.



- The two primary fields for exploring data in Looker are **dimensions** and **measures**.
- **Dimensions** are attributes or characteristics of your data. Each column in a database table is a dimension in Looker.



## Exploring Data

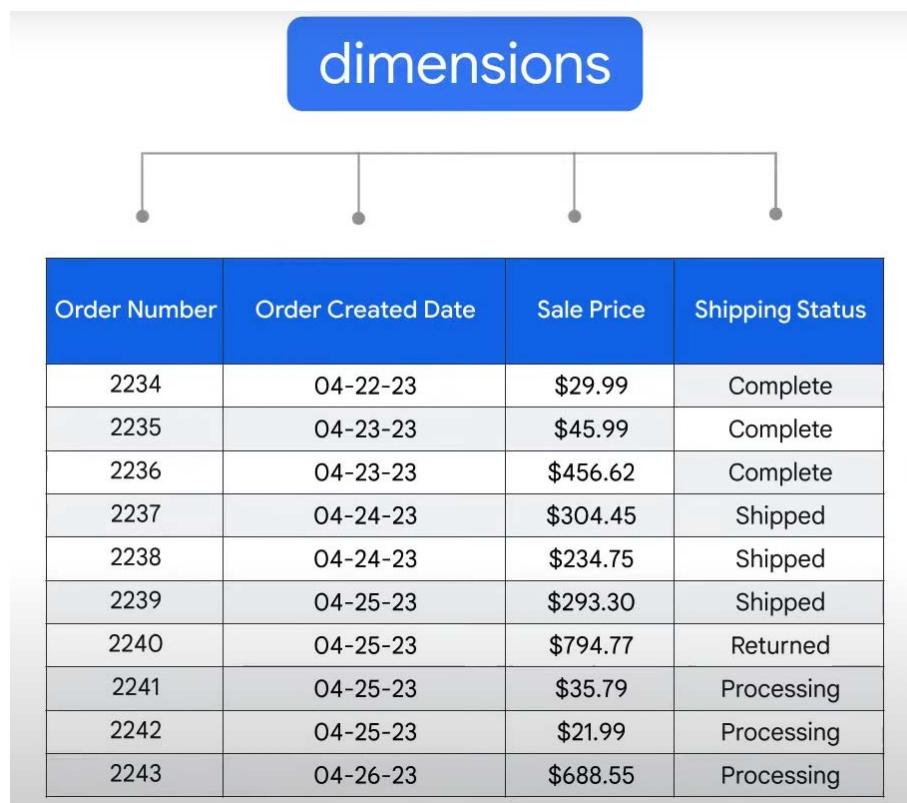
In this example, we will work with **ecommerce data**.

| Order Number | Order Created Date | Sale Price | Shipping Status |
|--------------|--------------------|------------|-----------------|
| 2234         | 04-22-23           | \$29.99    | Complete        |
| 2235         | 04-23-23           | \$45.99    | Complete        |
| 2236         | 04-23-23           | \$456.62   | Complete        |
| 2237         | 04-24-23           | \$304.45   | Shipped         |
| 2238         | 04-24-23           | \$234.75   | Shipped         |
| 2239         | 04-25-23           | \$293.30   | Shipped         |
| 2240         | 04-25-23           | \$794.77   | Returned        |
| 2241         | 04-25-23           | \$35.79    | Processing      |
| 2242         | 04-25-23           | \$21.99    | Processing      |
| 2243         | 04-26-23           | \$688.55   | Processing      |

Each column in the table displays an attribute for the order, such as the date the order was placed, the sale price, and the shipping status.

### 1. Dimensions:

- In Looker, all of these columns are dimensions.

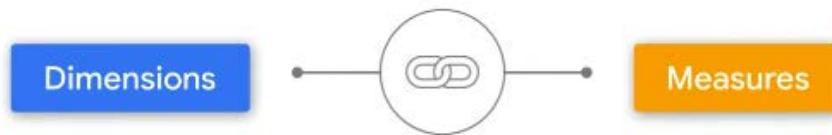


### 2. Measures:

- **Measures** are calculations performed across multiple rows of data, typically aggregates of data attributes or dimensions.
- Perhaps you want to learn how many orders were placed last week. Anytime you have a question like “how many,” you probably want to look for a measure with the word count or number in it (in this case, count). To find the answer to this data question, you must first group the data by dimension—in this case, the order date and then add a count measure.

| Order Items Order Count | Count |
|-------------------------|-------|
| 1                       | 1     |
| 2                       | 170   |
| 3                       | 237   |
| 4                       | 188   |
| 5                       | 306   |
| 6                       | 311   |
| 7                       | 318   |
| 8                       | 318   |
| 9                       | 375   |
| 10                      | 225   |
| 11                      | 219   |
| 12                      | 278   |
| 13                      | 292   |
| 14                      | 322   |
| 15                      | 331   |
| 16                      | 360   |
| 17                      | 228   |
| 18                      | 214   |
| 19                      | 270   |
| 20                      | 309   |

### 3. Combining Dimensions and Measures:



- Often, you need to combine dimensions and measures to answer your data questions. These can also span across multiple views.

## Analyzing User Data

### 1. Finding User Insights:

- To analyze user data, dimensions in the field picker are listed alphabetically, and some are grouped (e.g., Created Date).
- Use the search function to quickly locate dimensions. For example, if you want to identify the cities with the most users, you would add the **city dimension** and a **count measure**.

- Use the search function to quickly find the dimension you're looking for. In this example, the goal is to identify which **cities have the most users**, so start by adding the **city dimension** and then the count measure to see how many **users are in each city**.

The screenshot shows the Looker Explore interface with the 'Order Items' dataset selected. The left sidebar shows various fields under 'All Fields'. The 'Dimensions' section is expanded, and the 'City' field is highlighted with a green circle containing the number 1. The right panel shows a visualization titled 'Users City' with a single row labeled 'Users City'. The status bar at the bottom indicates 'Will process 3.25 MB'.

The screenshot shows the Looker Explore interface with the 'Order Items' dataset selected. The left sidebar shows various fields under 'All Fields'. The 'Measures' section is expanded, and the 'Count' measure is highlighted with a green circle containing the number 2. The right panel shows a visualization titled 'Users Count' with a single row labeled 'Users Count'. The status bar at the bottom indicates 'Will process 3.25 MB'.

- Set a row limit (e.g., to 10) to see the top 10 cities.

The screenshot shows the Looker Explore interface. On the left, there's a sidebar titled "Order Items" with a search bar "Search Fields Below". Under "All Fields", there are several dropdown menus: Age, City, Country, Created Date, Email, First Name, Gender, ID, Last Name, Latitude, Longitude, State, Traffic Source, and Zip. Under "MEASURES", there is a single "Count" option. On the right, there's a "Filters" section with a dropdown menu. Below it is a "Visualization" section with tabs for "Data", "Results", and "SQL". The "Data" tab is selected, showing a table with one row labeled "Users City". The table has two columns: "Users City" and "Users Count". The data is sorted by "Users Count" in descending order. The first ten rows are listed:

| Users City     | Users Count |
|----------------|-------------|
| 1 New York     | 3,369       |
| 2 Los Angeles  | 1,518       |
| 3 Chicago      | 1,058       |
| 4 Houston      | 909         |
| 5 Philadelphia | 642         |
| 6 Phoenix      | 594         |
| 7 San Diego    | 550         |
| 8 San Antonio  | 522         |
| 9 Dallas       | 492         |
| 10 Columbus    | 446         |

A green circle with the number "3" is overlaid on the "Run" button in the top right corner.

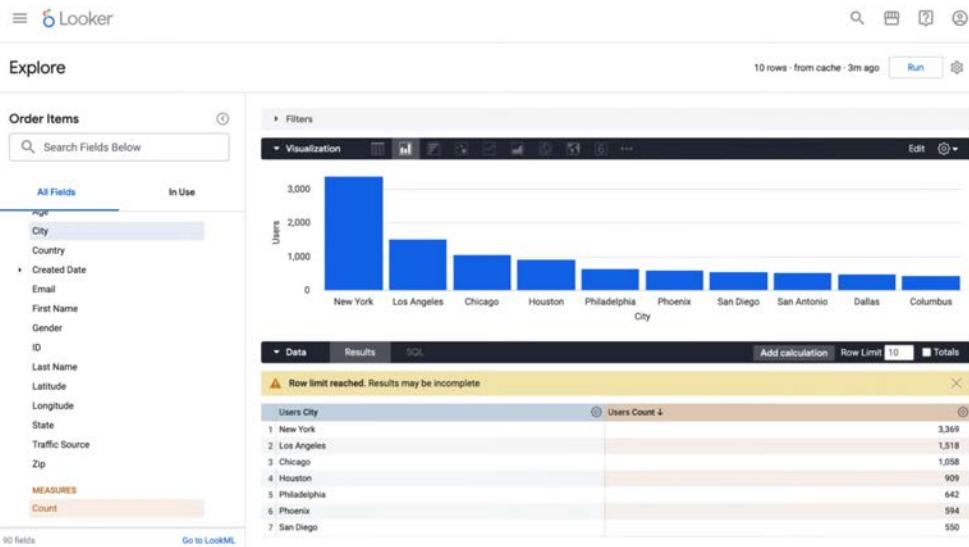
- Click **Run** to display the results.

This screenshot shows the same Looker Explore interface as the previous one, but the "Results" tab is now selected in the "Visualization" section. The results table now displays 10 rows of data, each showing a city name and its corresponding user count. The table is sorted by user count in descending order. The green circle with the number "3" is still overlaid on the "Run" button.

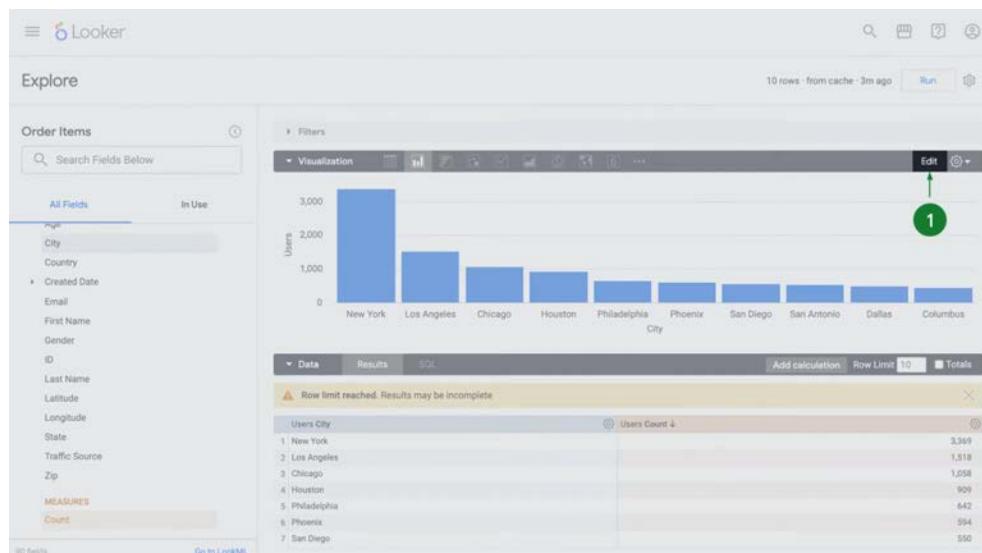
| Users City     | Users Count |
|----------------|-------------|
| 1 New York     | 3,369       |
| 2 Los Angeles  | 1,518       |
| 3 Chicago      | 1,058       |
| 4 Houston      | 909         |
| 5 Philadelphia | 642         |
| 6 Phoenix      | 594         |
| 7 San Diego    | 550         |
| 8 San Antonio  | 522         |
| 9 Dallas       | 492         |
| 10 Columbus    | 446         |

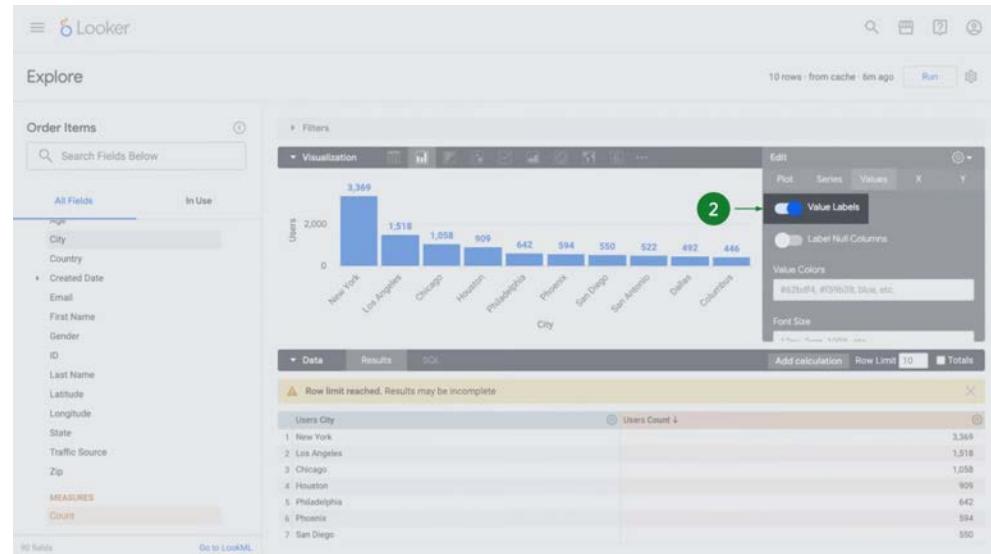
## 2. Adjusting Visualization:

- Looker offers various visualization options. For this example, a **column chart** is appropriate.

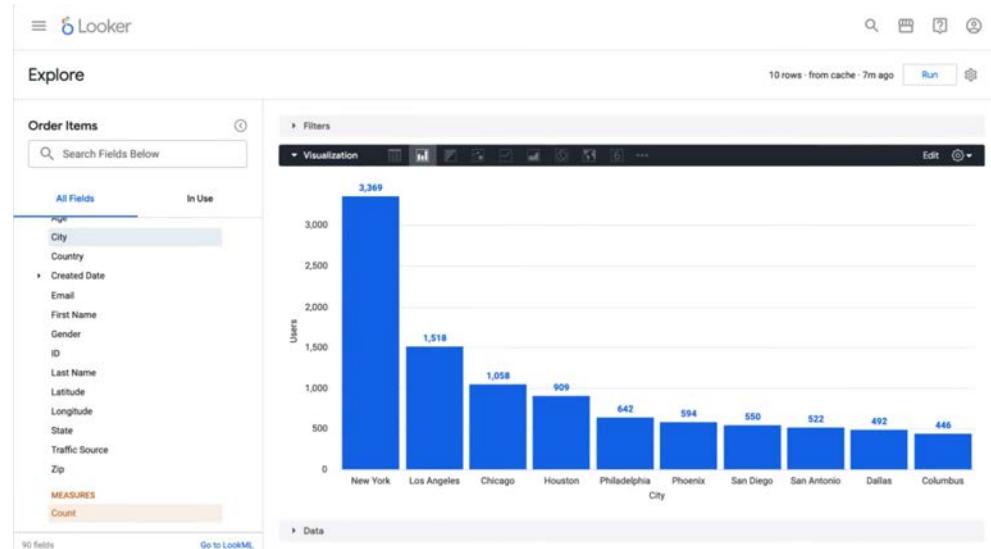


- To add value labels to the bars, click **Edit** and enable **Value Labels** in the **Values tab**.



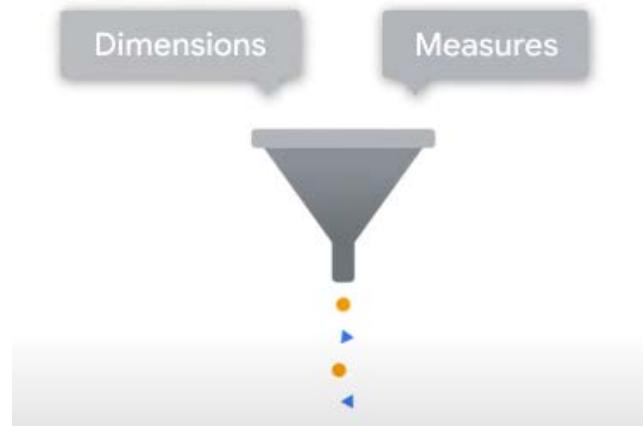


- Each column in the chart now has its value displayed.

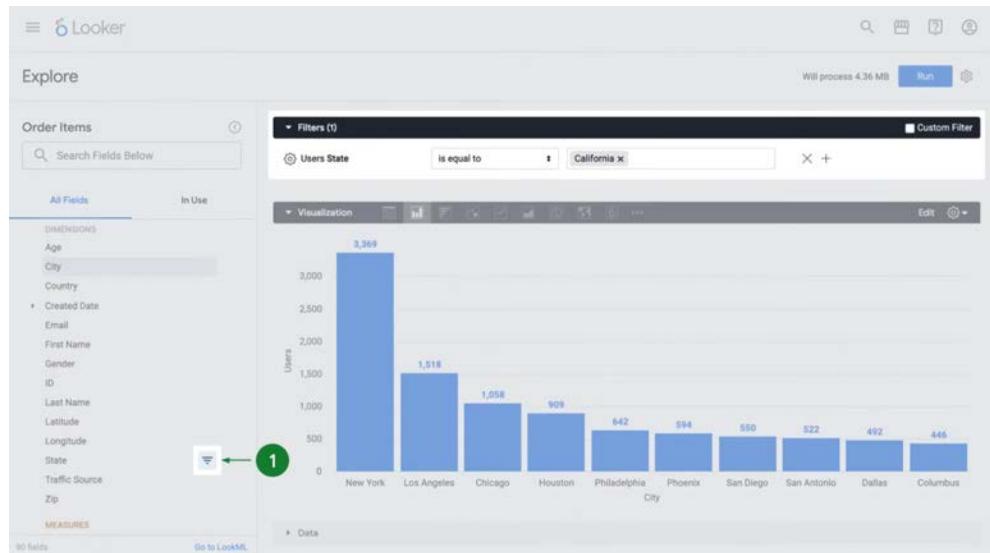


### 3. Filtering Results:

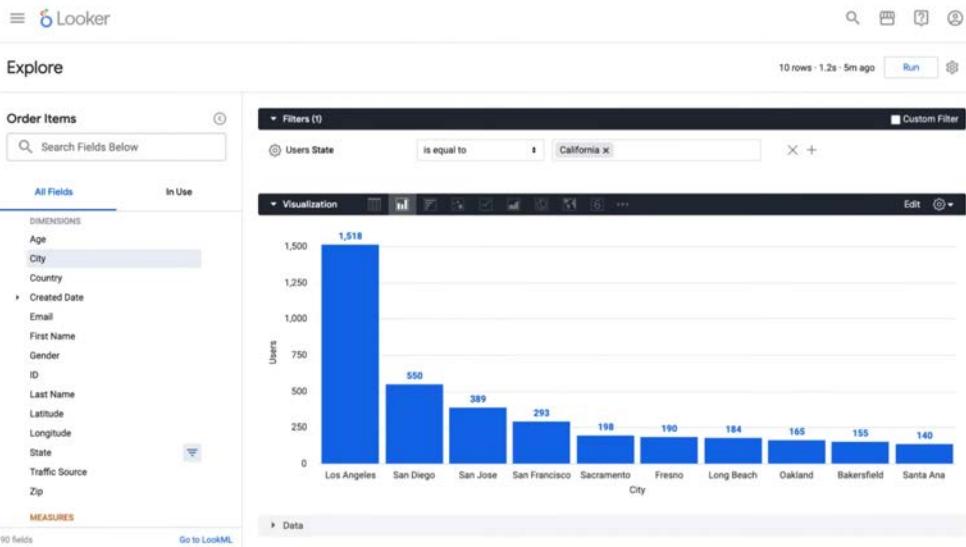
- Filters narrow the results based on specific criteria without deleting anything from the database.
- A key feature of filters is that they don't delete anything from the database; they're only applied to the data that Looker displays on your screen.
- You can filter both dimensions and measures.



- For instance, to focus on cities in **California**, you would add a filter for the state and rerun the query.



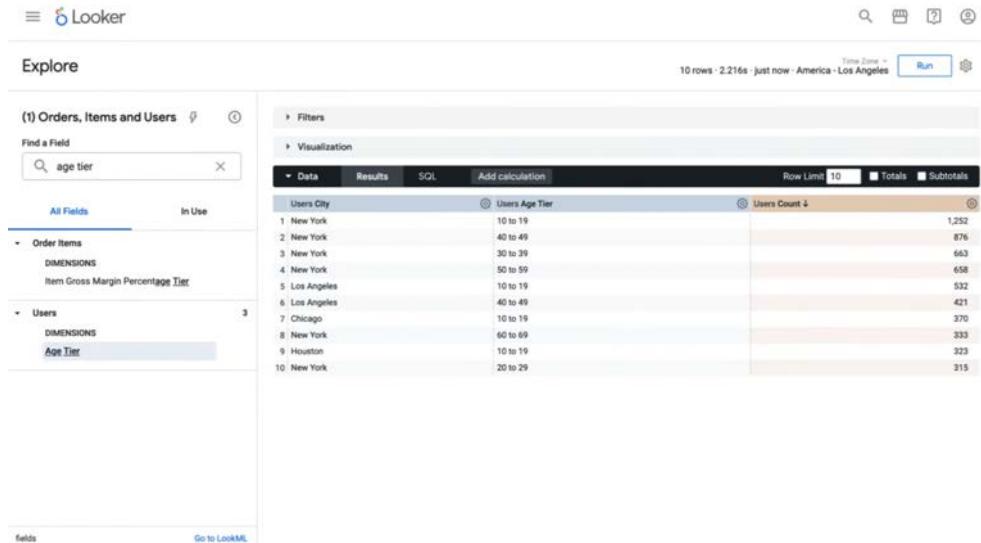
- After you rerun the query, The results are filtered to display just the top 10 cities in California.



## Pivoting Data

### 1. Understanding Pivoting:

- **Pivots** allow you to turn a selected dimension into several columns, creating a matrix of your data.
- For example, if you add the **age tier dimension**, you can group users by age.



### 2. Applying Pivoting:

- After running the query, if cities appear on multiple rows, pivoting on the **Age Tier** dimension will show all age tiers and cities visibly in a single view.
- Each unique value of the pivot dimension becomes its own column header.

The screenshot shows the Looker Explore interface with the following details:

- Explore Title:** (1) Orders, Items and Users
- Find a Field:** age tier
- Dimensions:** Order Items, Users (selected), Users Age Tier
- Measures:** Item Gross Margin Percentage, Tier
- Visualizations:** A pivot table titled "Users Age Tier".
- Data Headers:** Users City, Users Count, Users Count.
- Data Rows (Top 10):**

| Users City      | Users Count |
|-----------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| 1. New York     | 1,232       | 215         | 663         | 876         | 658         | 333         | 306         |             |             |             |
| 2. Los Angeles  | 532         | 144         | 298         | 421         | 274         | 146         | 129         |             |             |             |
| 3. Chicago      | 370         | 105         | 212         | 298         | 205         | 106         | 98          |             |             |             |
| 4. Houston      | 323         | 86          | 162         | 240         | 146         | 82          | 61          |             |             |             |
| 5. Philadelphia | 235         | 59          | 139         | 149         | 129         | 75          | 63          |             |             |             |
| 6. Phoenix      | 232         | 65          | 110         | 164         | 113         | 51          | 36          |             |             |             |
| 7. San Antonio  | 204         | 67          | 98          | 163         | 105         | 49          | 44          |             |             |             |
| 8. San Diego    | 200         | 33          | 87          | 116         | 112         | 47          | 39          |             |             |             |
| 9. Dallas       | 174         | 56          | 102         | 111         | 83          | 51          | 45          |             |             |             |
| 10. Columbus    | 162         | 51          | 80          | 109         | 78          | 53          | 33          |             |             |             |

### 3. Example of Pivoting:

- In the **Order Items Explore**, you can add the **Total Sale Price** measure along with the **Product Category** and **Department** dimensions.

The screenshot shows the Looker Explore interface with the following details:

- Explore Title:** Order Items
- Find a Field:** total sale
- Dimensions:** Order Items
- Measures:** Returned Total Sale Price, Total Sale Price
- Visualizations:** A pivot table titled "Order Items Total Sale Price".
- Data Headers:** Order Items Total Sale Price.
- Data Rows (Top 10):**

| Order Items Total Sale Price      |
|-----------------------------------|
| Press "Run" to explore this data. |

- Next, the Product Category and Department dimensions are added to group the results by these characteristics.

The screenshot shows the Looker Explore interface. On the left, the 'Order Items' model is selected, and the 'department' field is highlighted in the search bar. The sidebar shows the 'Products' dimension expanded, with 'Department' selected. The main area displays a visualization with a single row for 'Products Category' (Outerwear & Coats) and 'Products Department' (Men). A message at the bottom says 'Press "Run" to explore this data.'

- After the query is run, the two departments appear on multiple rows.

The screenshot shows the same Looker Explore interface after the query has been run. The results table now contains 20 rows, each corresponding to a product category and its gender. The columns are 'Products Category' (e.g., Outerwear & Coats, Jeans, Suits & Sport Coats, etc.) and 'Products Department' (Men or Women), along with the calculated column 'Order Items Total Sale Price'. The data shows various items like Men's Outerwear & Coats (\$868,731.90), Men's Jeans (\$799,435.90), and Women's Dresses (\$475,140.46).

| Products Category                 | Products Department | Order Items Total Sale Price |
|-----------------------------------|---------------------|------------------------------|
| 1: Outerwear & Coats              | Men                 | \$868,731.90                 |
| 2: Jeans                          | Men                 | \$799,435.90                 |
| 3: Suits & Sport Coats            | Men                 | \$644,334.48                 |
| 4: Sweaters                       | Men                 | \$547,082.03                 |
| 5: Jeans                          | Women               | \$475,140.46                 |
| 6: Dress                          | Women               | \$470,423.22                 |
| 7: Outerwear & Coats              | Women               | \$466,988.48                 |
| 8: Intimates                      | Women               | \$452,303.09                 |
| 9: Pants                          | Men                 | \$418,408.10                 |
| 10: Fashion Hoodies & Sweatshirts | Men                 | \$395,147.22                 |
| 11: Swim                          | Women               | \$332,513.74                 |
| 12: Shorts                        | Men                 | \$329,432.09                 |
| 13: Sleep & Lounge                | Men                 | \$325,399.32                 |
| 14: Tops & Tees                   | Men                 | \$322,206.19                 |
| 15: Swim                          | Men                 | \$314,194.31                 |
| 16: Blazers & Jackets             | Women               | \$295,205.61                 |
| 17: Sweaters                      | Women               | \$293,105.51                 |
| 18: Maternity                     | Women               | \$255,946.07                 |
| 19: Active                        | Men                 | \$252,718.60                 |
| 20: Fashion Hoodies & Sweatshirts | Women               | \$249,741.15                 |

- Pivoting on the **department** dimension brings department to the top of the table, so when you rerun the query, each department is displayed in a single column.

Looker

Explore

Order Items

Find a Field

All Fields In Use

- Products
- DIMENSIONS
- Department

Fields Go to LookML

Filters

Visualization

Data Results SQL Add calculation

Row Limit 500 Totals Subtotals

Time Zone Run

36 rows · from cache · 3m ago · America - Los Angeles

| Products Category                | Products Department | Order items Total Sale Price |
|----------------------------------|---------------------|------------------------------|
| 1 Outerwear & Coats              | Men                 | \$868,731.90                 |
| 2 Jeans                          | Men                 | \$799,435.90                 |
| 3 Suits & Sport Coats            | Men                 | \$644,634.48                 |
| 4 Sweaters                       | Men                 | \$547,082.03                 |
| 5 Pants                          | Men                 | \$475,140.46                 |
| 6 Dress                          | Women               | \$470,425.22                 |
| 7 Outerwear & Coats              | Women               | \$466,988.48                 |
| 8 Intimates                      | Women               | \$452,503.09                 |
| 9 Panta                          | Men                 | \$418,408.10                 |
| 10 Fashion Hoodies & Sweatshirts | Men                 | \$399,147.22                 |
| 11 Swim                          | Women               | \$332,513.74                 |
| 12 Shorts                        | Men                 | \$329,432.09                 |
| 13 Sleep & Lounge                | Men                 | \$325,399.32                 |
| 14 Tops & Tees                   | Men                 | \$322,208.19                 |
| 15 Swim                          | Men                 | \$316,194.31                 |
| 16 Blazers & Jackets             | Women               | \$295,205.61                 |
| 17 Sweaters                      | Women               | \$293,105.51                 |
| 18 Maternity                     | Women               | \$255,560.07                 |
| 19 Active                        | Men                 | \$252,718.60                 |
| 20 Fashion Hoodies & Sweatshirts | Women               | \$249,741.15                 |

Looker

Explore

Order Items

Find a Field

All Fields In Use

- Products
- DIMENSIONS
- Department

Fields Go to LookML

Filters

Visualization

Data Results SQL Add calculation

Row Limit 500 Column Limit 50 Totals Row Totals

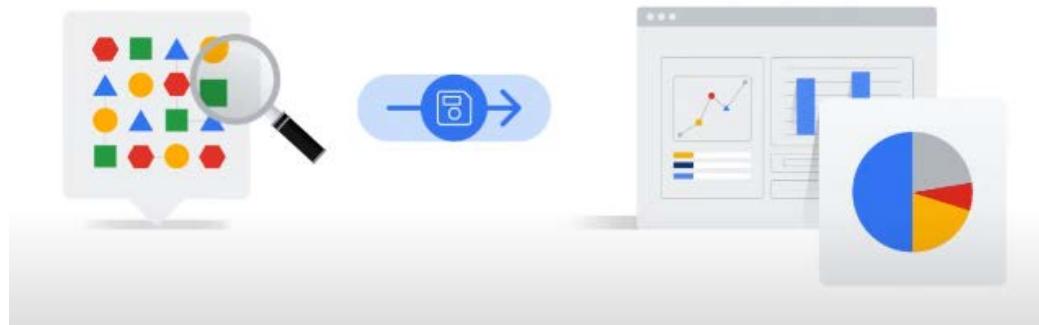
Time Zone Run

26 rows · 3.395s · just now · America - Los Angeles

| Products Category               | Men          | Women        | Order items Total Sale Price |
|---------------------------------|--------------|--------------|------------------------------|
| 1 Outerwear & Coats             | \$868,731.90 | \$466,988.48 |                              |
| 2 Jeans                         | \$799,435.90 | \$475,140.46 |                              |
| 3 Suits & Sport Coats           | \$644,634.48 |              |                              |
| 4 Sweaters                      | \$547,082.03 |              | \$293,105.51                 |
| 5 Pants                         | \$475,140.46 |              |                              |
| 6 Fashion Hoodies & Sweatshirts | \$418,408.10 |              |                              |
| 7 Shorts                        | \$399,147.22 |              |                              |
| 8 Sleep & Lounge                | \$332,513.74 |              |                              |
| 9 Tops & Tees                   | \$329,432.09 |              |                              |
| 10 Swim                         | \$325,399.32 |              |                              |
| 11 Active                       | \$322,208.19 |              |                              |
| 12 Accessories                  | \$316,194.31 |              |                              |
| 13 Underwear                    | \$295,205.61 |              |                              |
| 14 Socks                        | \$252,718.60 |              |                              |
| 15 Dresses                      | \$255,560.07 |              |                              |
| 16 Intimates                    | \$255,888.25 |              |                              |
| 17 Blazers & Jackets            | \$209,700.98 |              |                              |
| 18 Maternity                    | \$183,234.74 |              |                              |
| 19 Pants & Capris               | \$124,395.67 |              |                              |

## Saving and Sharing Insights

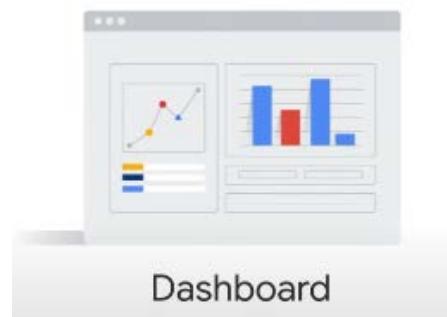
### 1. Looks and Dashboards:



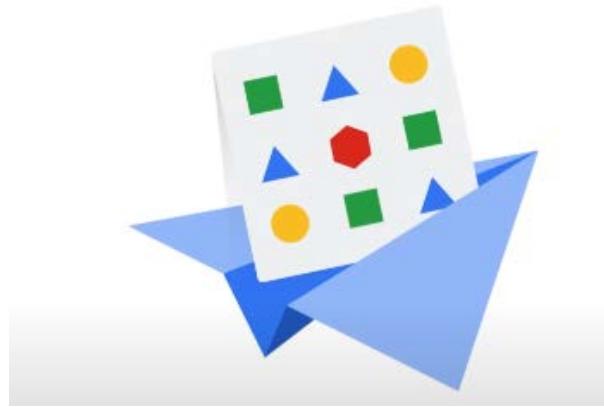
- If you discover interesting insights, you can save your findings as a **Look**, which is a single, saved report or visualization.



- Looks can be added to a **dashboard**, a series of saved reports on one page that presents answers to multiple data questions.



## 2. Sharing Data Findings:



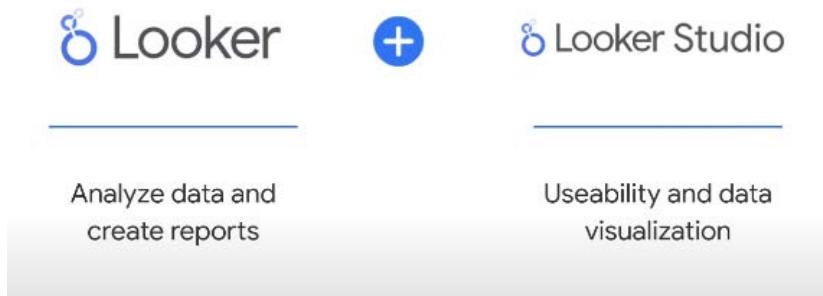
- Explores, Looks, and dashboards can all be shared with others, enabling collaboration and informed decision-making.

This overview provides a foundational understanding of how to analyze and visualize data using Looker, empowering you to derive insights and share findings effectively.

[https://www.cloudskillsboost.google/parts/18/course\\_templates/578/video/507303](https://www.cloudskillsboost.google/parts/18/course_templates/578/video/507303)

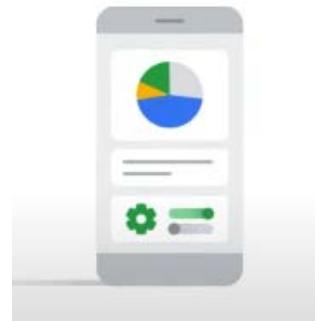
## ▼ Visualize Data Using Looker Studio

In addition to Looker, you can use **Looker Studio** to visualize your data effectively.



### Overview of Looker Studio

- **Looker Studio** provides powerful analytics tools for analyzing data and creating reports, but it focuses more on a simple, easy-to-use interface for data visualization.
- Looker Studio offers several features that facilitate sharing and collaboration on reports:
  - Publish reports to the web.
  - Embed reports in other applications.



## Benefits of Using Looker Studio

### 1. User-Friendly Interface:

- A drag-and-drop interface that simplifies the creation of reports and dashboards.

### 2. Diverse Visualization Options:

- Various charts, graphs, and tables available for reporting.

### 3. Professional Features:

- Add filters and drill down into data to create professional-looking reports.

### 4. Real-Time Collaboration:

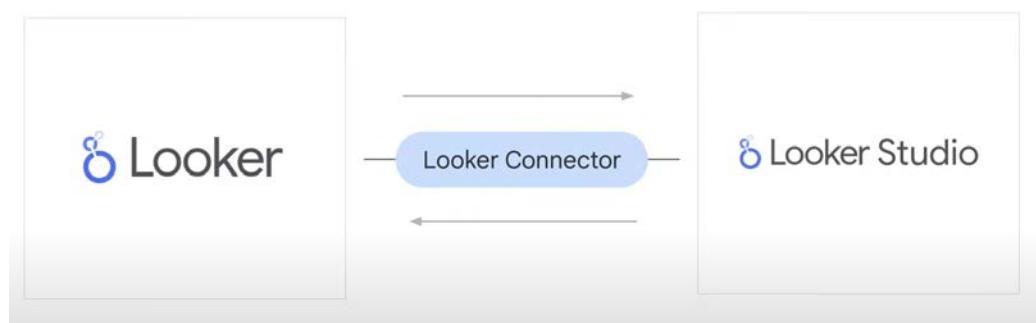
- Collaborate on reports and dashboards in real time.

### 5. Security Controls:

- Control who has access to your data through robust security features.

## Integration with Looker

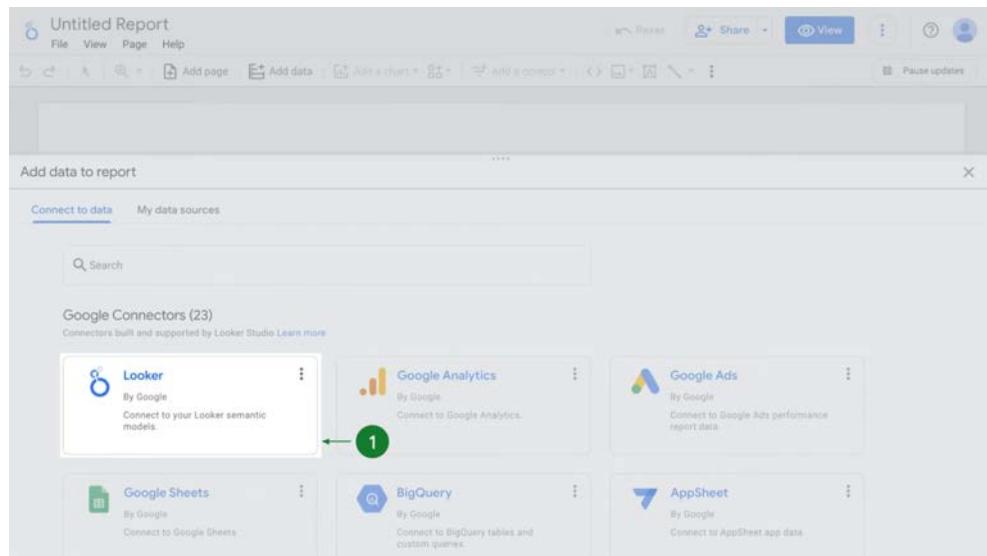
- Looker and Looker Studio can be integrated using the **Looker connector for Looker Studio**.



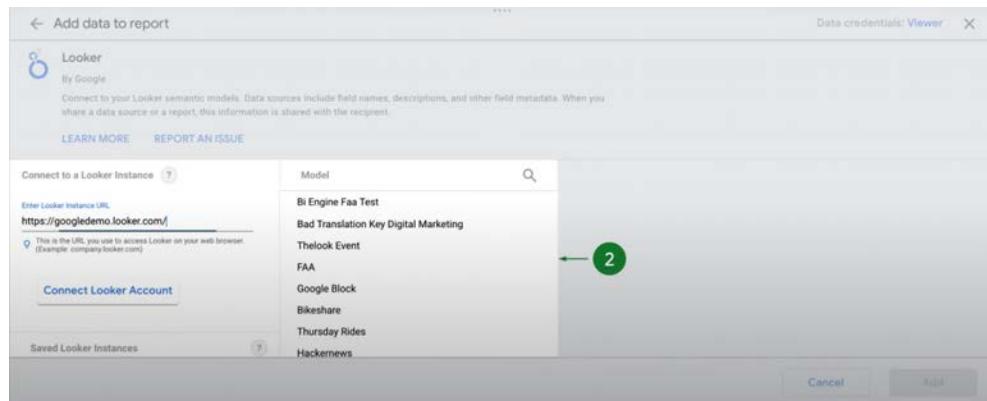
- This integration allows you to connect to any Google Cloud-hosted Looker instance and create reports and dashboards using Looker data.

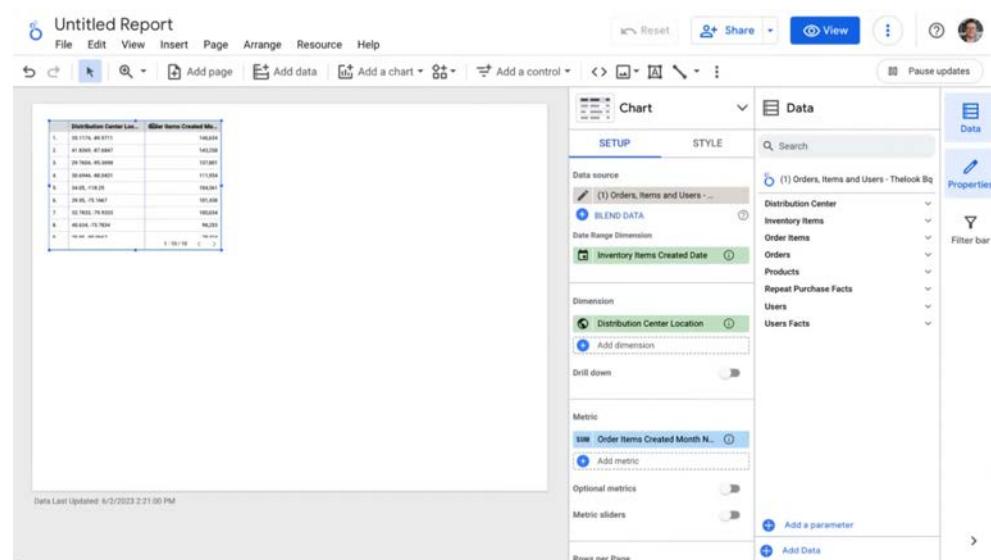
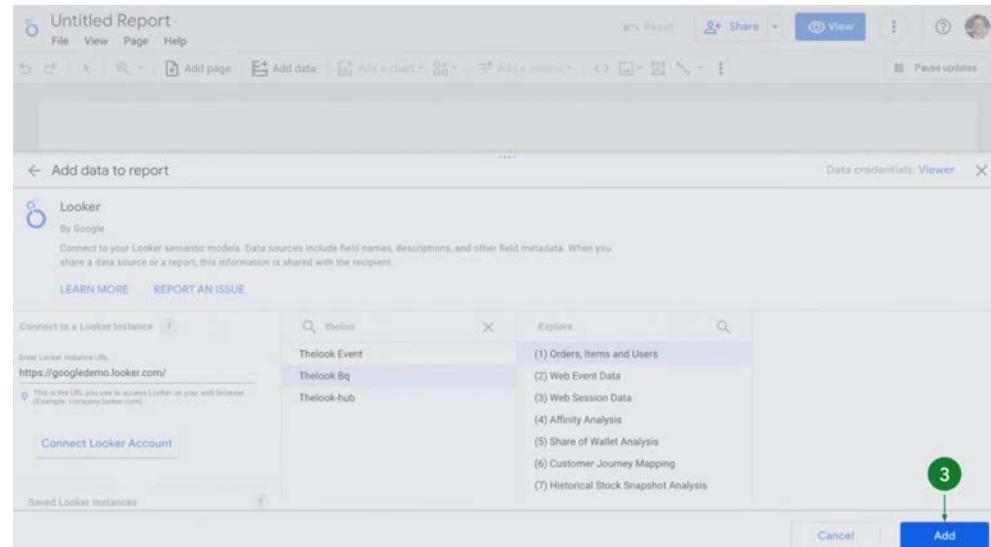
## Looker BI Connector Enabled

- To use the Looker connector, you must first enable the integration in your Looker instance. Once enabled:
  - Create a new report in Looker Studio.
  - Select the Looker connector as your data source.



- Enter the URL of your Looker instance and select an **Explore**.





## Creating Visualizations

To create effective visualizations in Looker Studio, consider the following:

### 1. Data Type:

- What type of data are you working with?

### 2. Insights:

- What insights are you trying to communicate? Your message will influence the visualization type.

### 3. Audience:

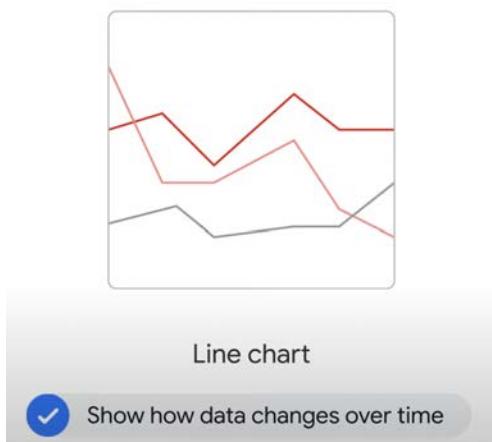
- Who is the report for? The audience may affect how you style and organize the report.

## Common Visualization Options

1. **Bar Charts:** Compare data across different categories.



2. **Line Charts:** Show how data changes over time.



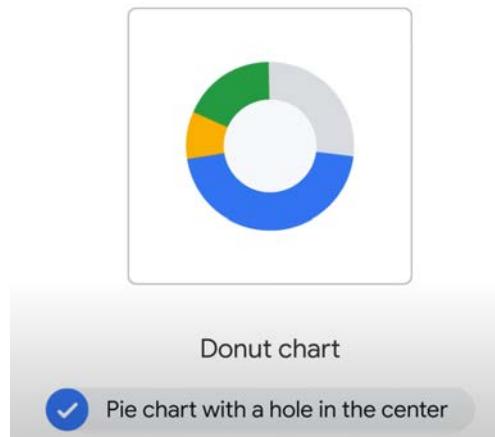
3. **Area Charts:** Similar to line charts but fill the area below the line.



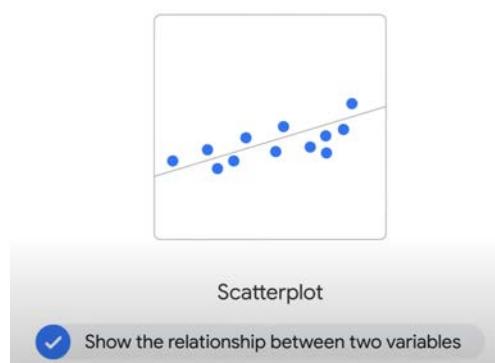
**4. Pie Charts:** Display relative proportions of different categories.



**5. Donut Charts:** Similar to pie charts, but with a hole in the center.



**6. Scatterplots:** Illustrate the relationship between two variables.



**7. Heatmaps:** Visualize the distribution of data.



Heatmap

Show the distribution of data

8. **Maps:** Show geographically located data.



Map

Show data that is geographically located

## Creating a Simple Chart in Looker Studio

Let's walk through creating a simple chart using Looker data:

### 1. Connect Looker as Data Source:

- Ensure Looker is connected using the Looker connector.

### 2. Understanding Reports:

- A report in Looker Studio is similar to a dashboard in Looker, consisting of one or several charts.

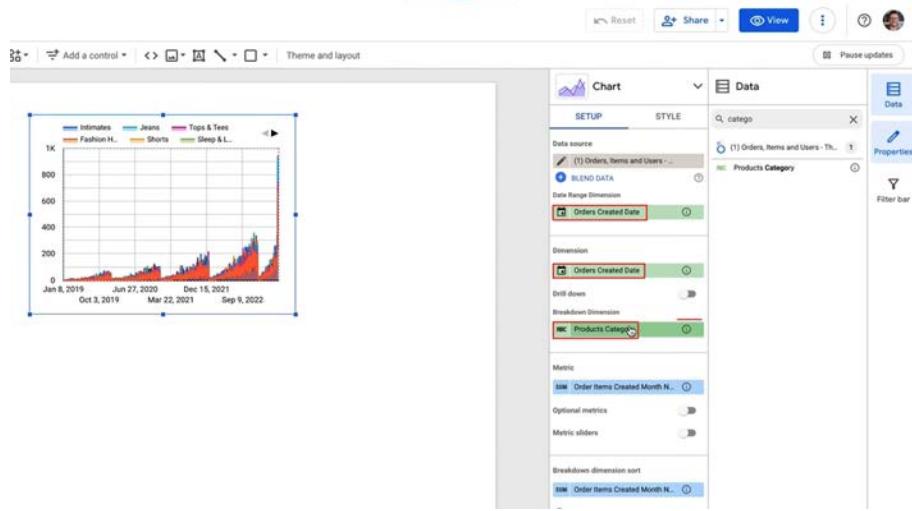
### 3. Adding a Chart:

- Click **Add a chart** in the toolbar and select the desired chart type.

- For example, if you work at an Ecommerce store and want to visualize the count of orders by category over the past year, a line chart or area chart would be ideal.

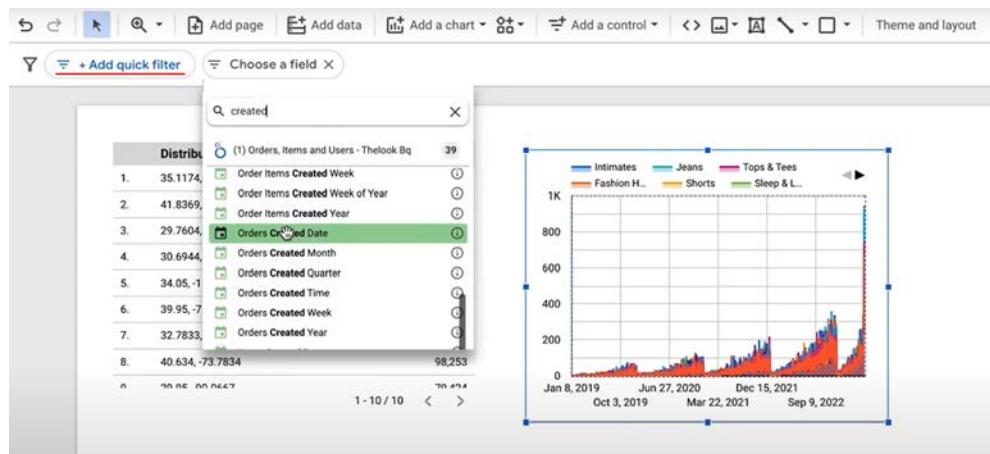
### 4. Customizing the Data:

- Add dimensions for **order created date** and **product category**.
- Use the **created date** as the date range dimension and specify **product category** as the breakdown dimension.

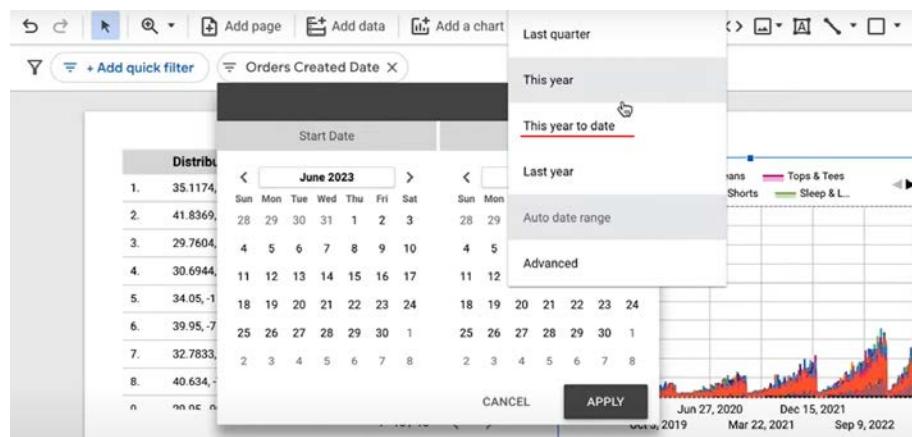


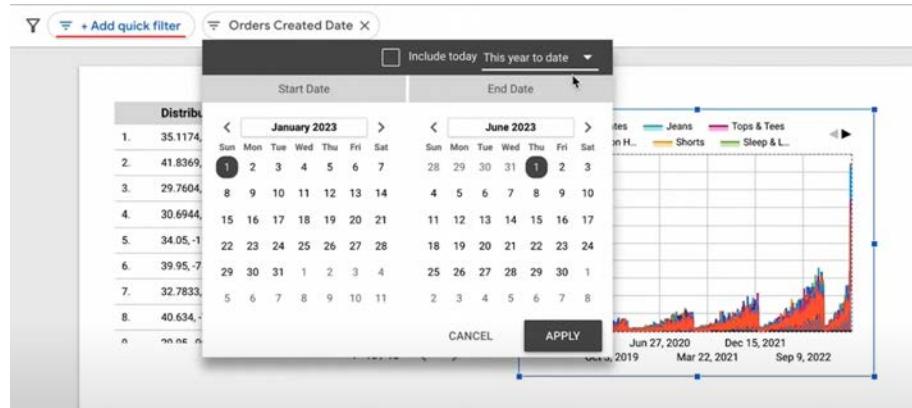
## 5. Filtering Results:

- To focus on the past year, add a filter to limit the results.



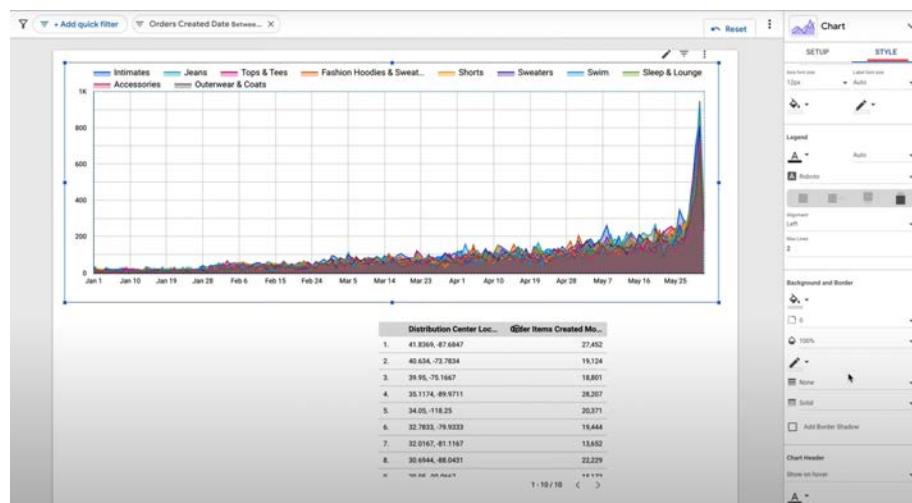
- Now, the chart displays the count of orders by date, broken down by category, for the past year.





## 6. Customizing the Chart:

- Under the **Style** tab, you can customize chart components such as labels, colors, and fonts or switch to a different visualization.



Using Looker Studio enhances your data visualization capabilities by providing a user-friendly interface, diverse visualization options, and integration with Looker. This enables effective communication of insights and collaboration on reports, helping you to make informed decisions based on your data.

## ▼ Sharing Reports and Visualizations in Looker and Looker Studio

The value of **Looker** and **Looker Studio** extends beyond data analysis and visualization; effective sharing of reports and dashboards enhances collaboration, promotes data-driven decision-making, and increases transparency.

### Data Sharing in Looker

### Benefits:



1. **Improved Collaboration:** Sharing data in Looker fosters better communication, leading to faster decision-making and increased efficiency.
2. **Centralized Repository:** Looker provides a centralized repository for data, ensuring that it is accurate, consistent, and easily accessible.
3. **Scalability:** Looker is scalable and can integrate with other data sources and applications.

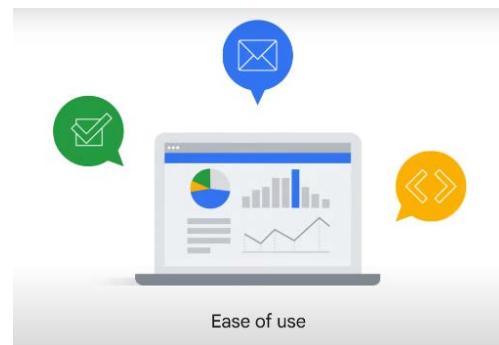
### Drawbacks:

1. **Security Concerns:** Storing data in the cloud can expose it to potential security breaches, although Looker provides security features such as encryption and access control to mitigate these risks.
2. **Cost:** Looker is a paid platform, which may be a consideration for some organizations.
3. **Complexity:** Looker can be relatively complex, presenting a steep learning curve for new users.

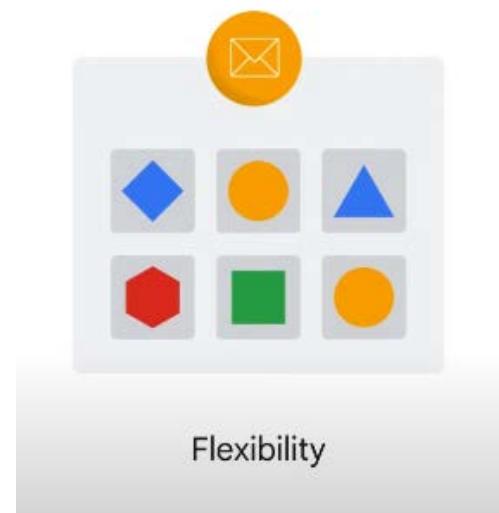
## Data Sharing in Looker Studio

### Benefits:

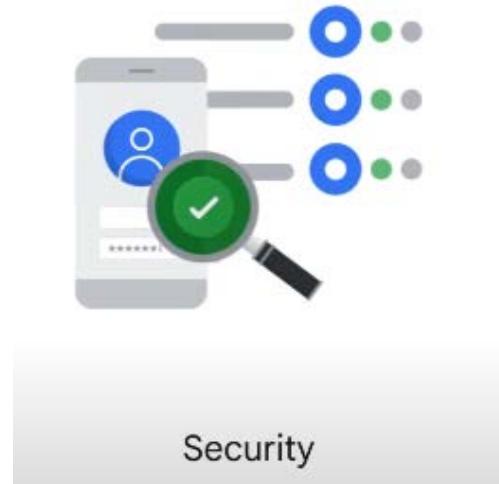
1. **Ease of Use:** Looker Studio is user-friendly, allowing users to easily create and share dashboards and reports, which can also be embedded in websites and applications.



2. **Flexibility:** It offers features that enable customization of the data-sharing experience, including access controls and permission settings.

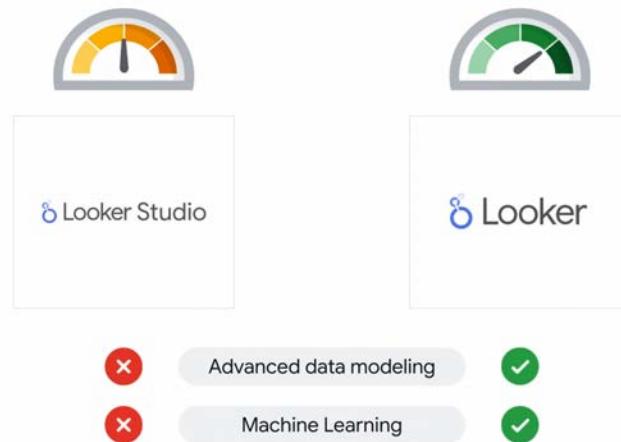


3. **Security Features:** Looker Studio includes security features like encryption and access control to protect shared data.

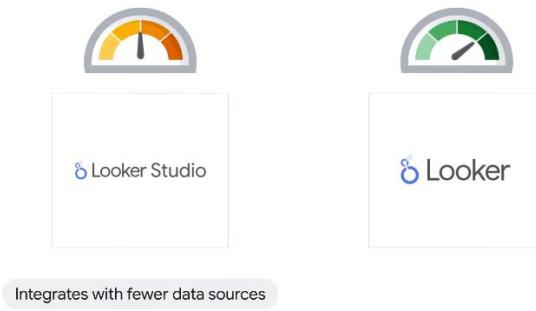


#### Drawbacks:

1. **Less Powerful:** Overall, Looker Studio is less powerful than Looker and lacks advanced features like data modeling and machine learning capabilities.



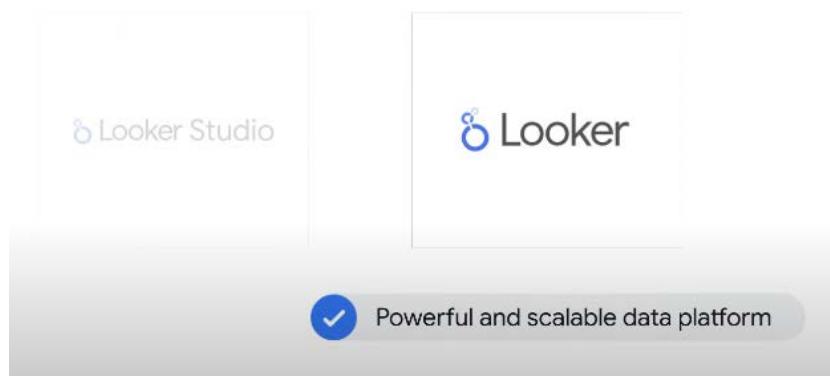
2. **Limited Integrations:** Looker Studio does not integrate with as many data sources as Looker, which can complicate data sharing from multiple sources.



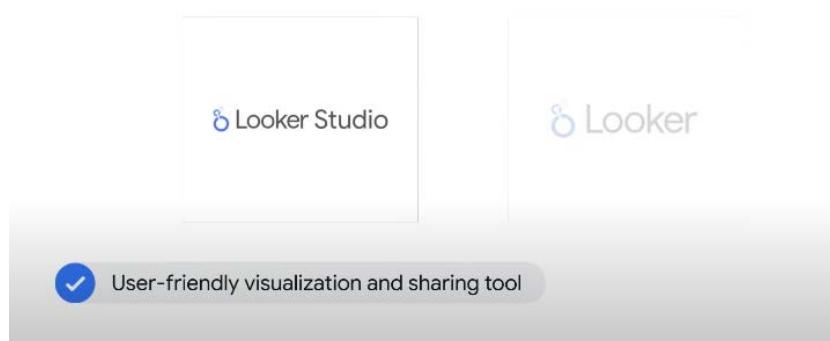
## Choosing the Right Platform

The choice between Looker and Looker Studio for data sharing depends on the specific needs of your organization:

- **Looker:** Best for organizations needing a powerful and scalable platform for comprehensive data sharing.



- **Looker Studio:** Ideal for organizations seeking a more user-friendly platform with straightforward sharing capabilities.

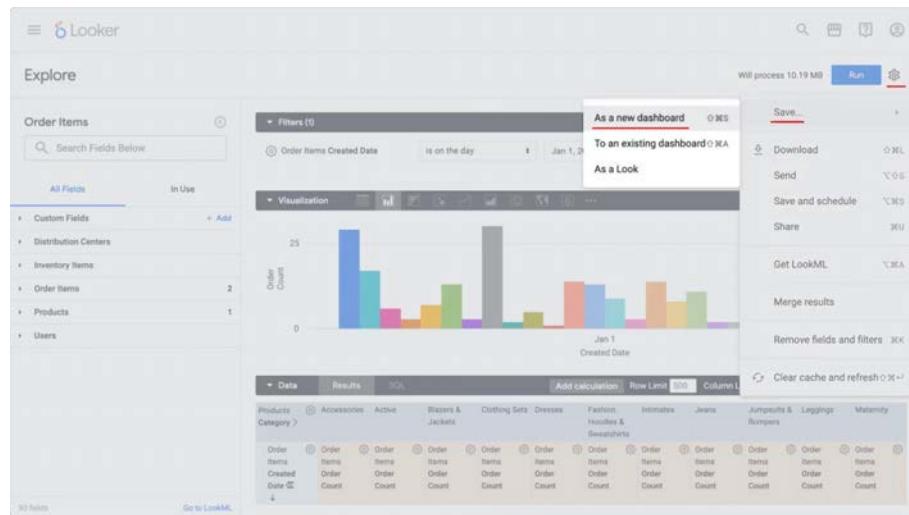


## How to Share Data in Looker

In Looker, sharing data is often referred to as "data delivery." You can share insights in several ways:

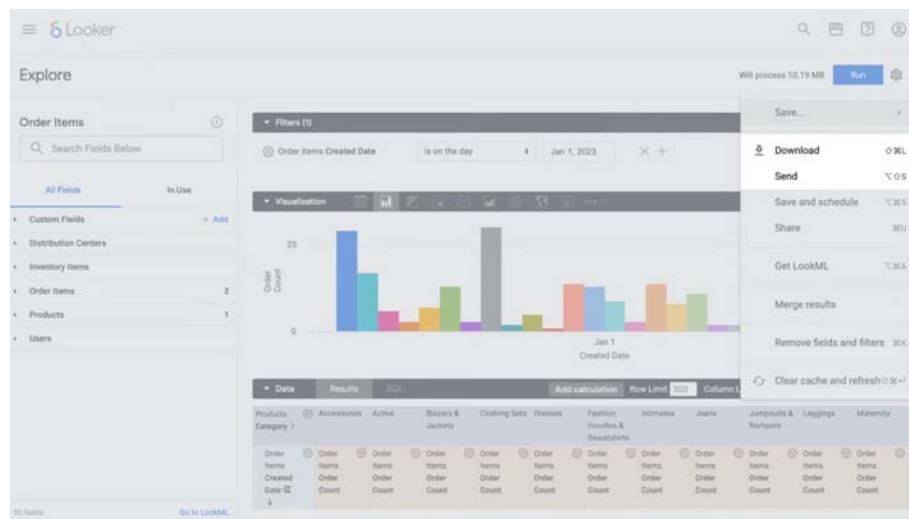
### 1. Save as a Report or Dashboard:

- Save your findings as a **Look** or to a dashboard displaying multiple reports.



### 2. Export Data:

#### a. Download:

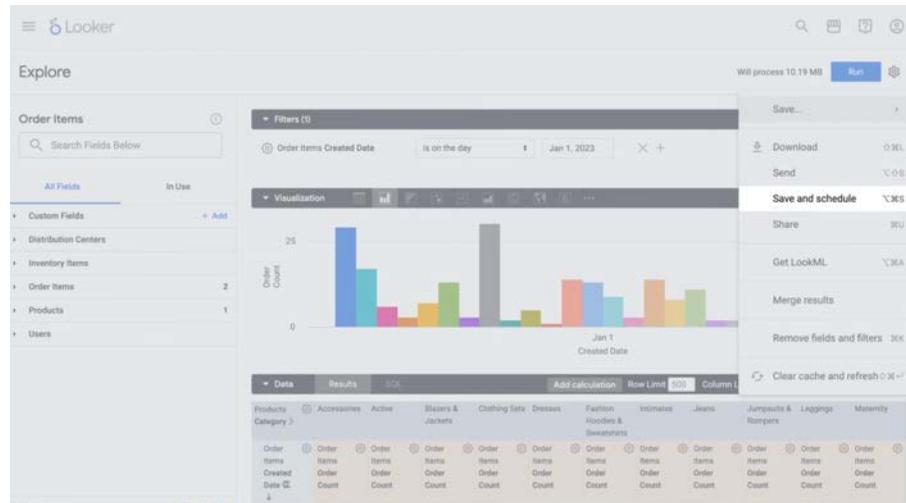


- Download data to your computer in various formats such as CSV, PDF, or PNG.
- For one-time exports, use the **Download** option.

## b. Send

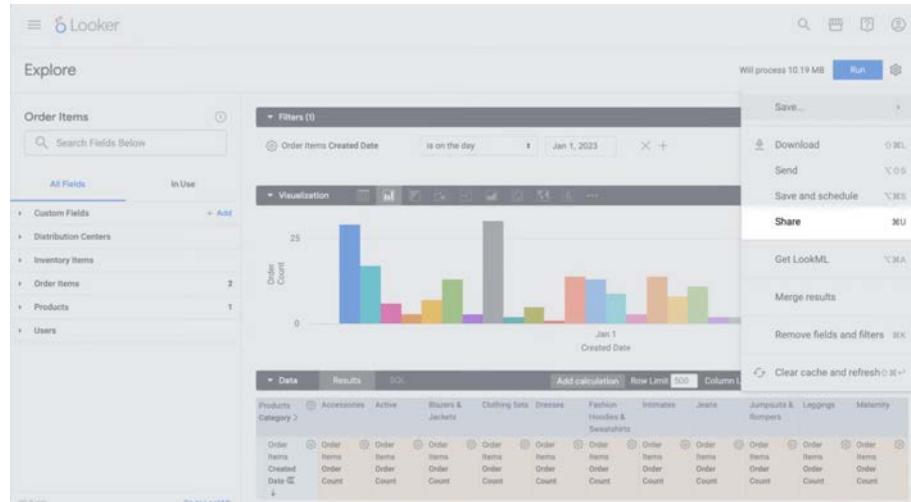
- Deliver data via email. For instance, you can email a dashboard to yourself and your manager.

## c. Save and Schedule

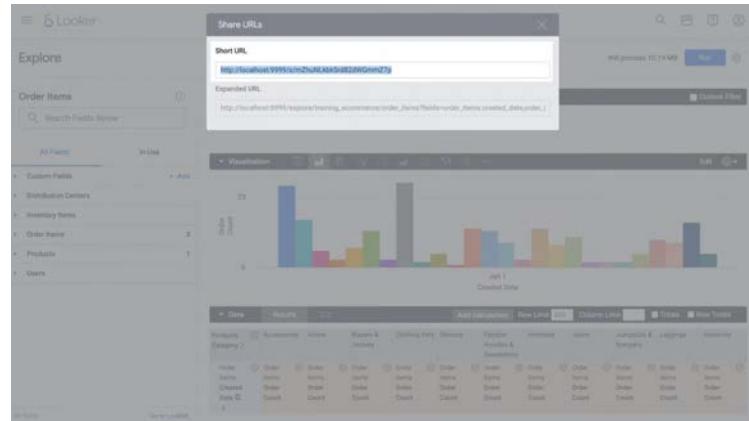


- Set up a **Schedule** to automate sending reports at defined intervals (e.g., daily or weekly).

## 3. Sharing Links:



- Click **Share**, copy the URL, and send it to others. Those with the appropriate permissions can view the data.



## Sharing Data in Looker Studio

Looker Studio allows data sharing through Google Drive, making it similar to sharing Google Docs or Sheets:



### 1. Permissions:

- Assign **edit** or **view permissions** to users when sharing reports and data sources.

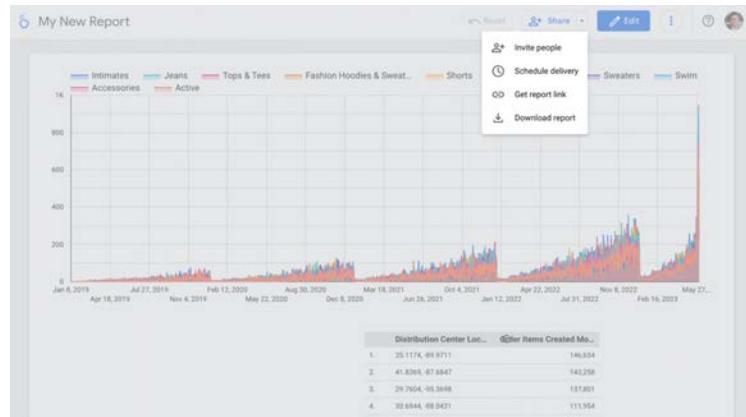


## 2. Sharing Reports:

- You can share a report directly from the Looker Studio **home page** or within the **report itself**.

The screenshot shows the Looker Studio Pro interface. At the top, there's a navigation bar with 'Create', 'Recent', 'Imports', 'Data sources', and 'Explorer'. Below it is a 'Recent' section with 'Shared with me' and 'Trash' options. To the right is a 'Template Gallery' with several templates like 'Blank Report', 'Tutorial Report', and 'Acme Marketing'. The main area displays a list of recent reports: 'My New Report' (owned by anyone), 'CIB YouTube Content', 'Looker Studio Reporting', 'Cloud Immersion Workshops', and 'Global Case Study Finder'. A context menu is open over 'My New Report', showing options: 'Share', 'Rename', 'Move to', and 'Remove'. The date 'May 2, 2022' is also visible.

- Options include scheduling delivery, obtaining a link to the report, or downloading it.



### 3. Send Reports:

- Specify whether recipients can view or edit the report by clicking **Send** to share.

Share with people and groups

Share as [REDACTED]

Add people and groups

**People with access**  
This content is managed by an organization and members within it

[REDACTED] Owner

Link settings

**Restricted** Only people added can open with this link

**Copy link** Done

← Share with people and groups

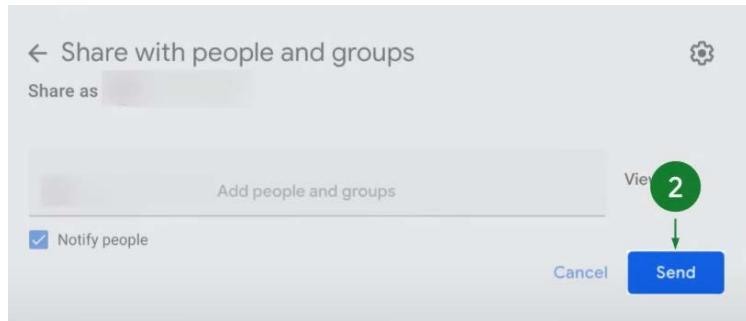
Share as [REDACTED]

Add people and groups

Notify people

**Viewer**

1 ✓ Viewer  
Editor



Both Looker and Looker Studio offer robust options for visualizing and sharing data. The best choice for you and your team will depend on your organization's specific needs, whether that's the powerful features of Looker or the user-friendly experience of Looker Studio.

## ▼ Summary

This module has equipped you with the knowledge to analyze, visualize, and share data using **Looker** and **Looker Studio**. Here's a recap of what you learned:

### 1. Data Exploration in Looker:

- You were introduced to Looker Explore, learning how to manipulate dimensions, measures, filters, and pivots to answer data-driven questions.

### 2. Visualization Options:

- You explored various visualization techniques available in both Looker and Looker Studio, and learned how to create charts in Looker Studio.

### 3. Sharing Data:

- You compared the pros and cons of using Looker and Looker Studio for sharing data and learned how to share reports and dashboards on both platforms.

You are now ready to utilize these tools to make informed, data-driven decisions.

## ▼ Module Summary

This concludes the **Introduction to Data Analytics on Google Cloud** course. Here's a quick recap of what you learned:

### 1. Foundational Concepts:

- The end-to-end data analytics lifecycle on Google Cloud, from data ingestion to analysis and activation.
- How to choose the appropriate data storage options and understand the different types of data used on Google Cloud.

## 2. BigQuery:

- How to use BigQuery at a high level to locate data for analysis.
- Basic SQL commands to answer data-driven questions.

## 3. Looker and Looker Studio:

- How to explore, analyze, and visualize data using Looker.
- The difference between a Look and a dashboard.
- When to use Looker or Looker Studio for data visualization and sharing.
- How to create visualizations and share reports on both platforms.

You are now equipped to start analyzing data using Google Cloud tools and processes.

# ▼ 2- BigQuery for Data Analysts

## ▼ 1- BigQuery for Data Analysts

### ▼ Data analytics on Google Cloud

As a data analyst, you're likely familiar with the challenges of working with large data sets. Data analysts, data engineers, and data scientists all analyze data, but they have distinct roles:

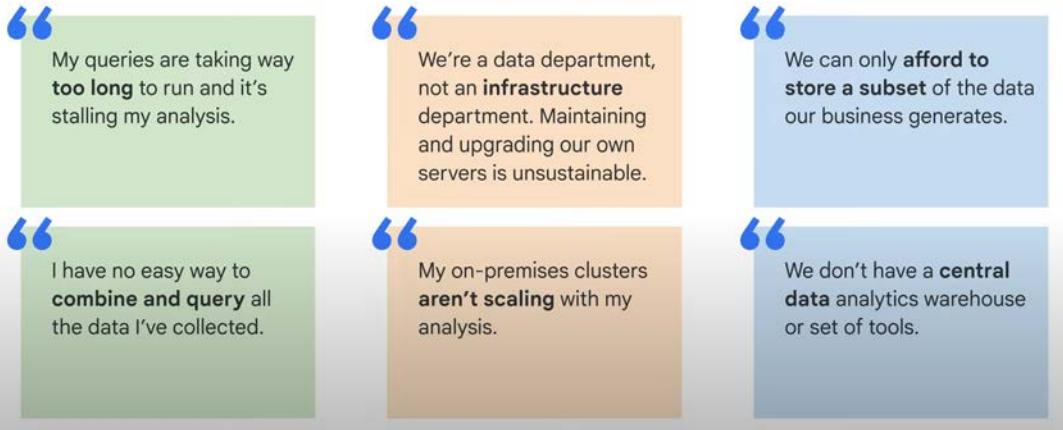
#### Data-specific roles

| Data analyst   | Data engineer  | Data scientist  |
|--|--|---|
| <ul style="list-style-type: none"><li>• What they do:<br/>Derive data insights from queries and visualization</li><li>• Background:<br/>Data analysis using SQL, UI tools, R, Python</li></ul> | <ul style="list-style-type: none"><li>• What they do:<br/>Design, build, and maintain data processing systems</li><li>• Background:<br/>Computer engineering</li></ul> | <ul style="list-style-type: none"><li>• What they do:<br/>Analyze data and model systems using statistics and machine learning</li><li>• Background:<br/>Statistical analysis using SQL, R, Python, Tensorflow, Pytorch, Keras, etc</li></ul> |

Although each role has different responsibilities, they often collaborate. For example, a data analyst may work with a data engineer to source and ingest data for analysis. In some cases, these roles can be performed by the same person, which is why understanding data ingestion and storage is crucial, even if you're focused on analysis.

## Common Challenges for Data Analysts

### Data analysts face query, infrastructure, and storage challenges



Two major challenges often arise for data analysts:

1. **Too much data** to process efficiently.
2. **Data that is not connected or centralized.**

These challenges can be broken down into three common themes:

- **Querying:** Long query execution times or overly complex logic.
- **Infrastructure:** Issues with scalability, performance, and maintenance.
- **Storage:** Limited capacity, high costs, or decentralized data.

These challenges are also faced by data engineers and data scientists. However, with Google Cloud, you can overcome these barriers, thanks to its scalable infrastructure that lets you focus on querying and analysis without worrying about storage and infrastructure limitations.

## Google Cloud for Data Analysis

Google Cloud offers solutions to storage and infrastructure challenges. It was designed with scalability in mind, enabling you to focus on your queries and gain insights faster. This is especially useful when comparing traditional on-premises infrastructure with cloud-based systems.

**01**

Storage is cheap and virtually unlimited  
in capacity

**02**

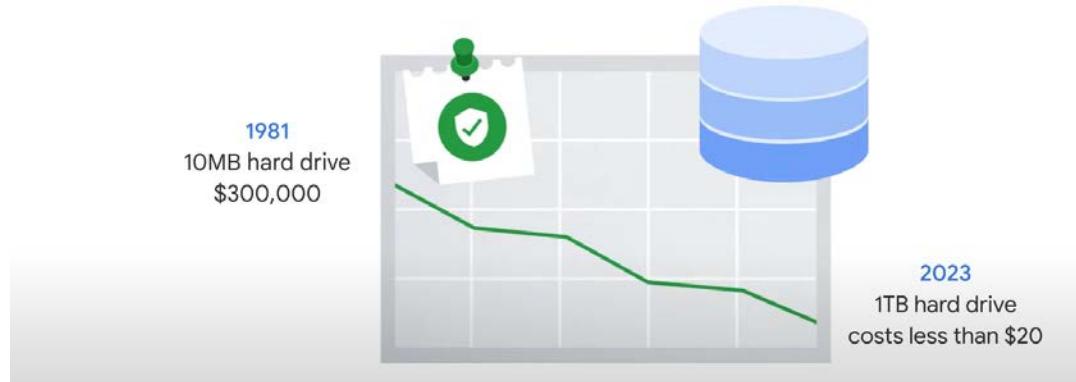
Focus on queries, not infrastructure

**03**

Massive scalability

Let's start with **storage**:

## The cost of storage has dropped dramatically

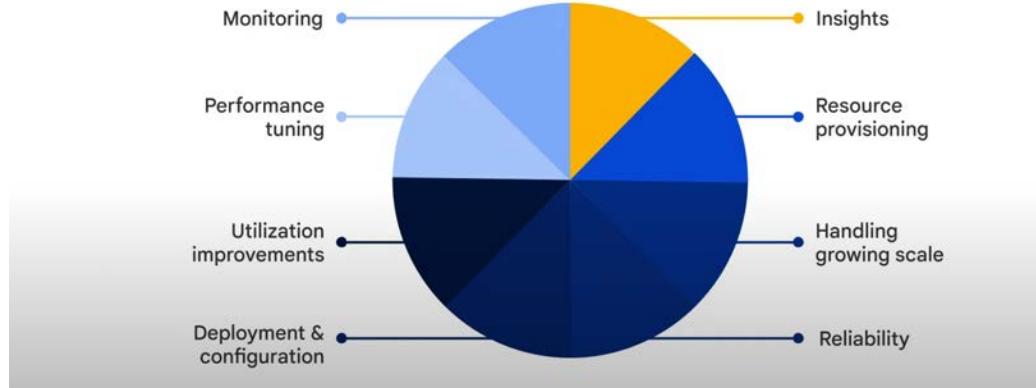


- In 1981, a 10 MB hard drive cost \$300,000. By 2023, a 1 TB hard drive costs less than \$20. While storage has become much cheaper, you also need:



- **Computing power**
- **Networking capabilities**
- **Admin and hardware teams** to maintain the infrastructure.
- **Software and licenses.**

## Typical big data processing



Managing all these components can be expensive and time-consuming, and traditional databases often can't handle the scale of modern data.

## Google Cloud's Advantages

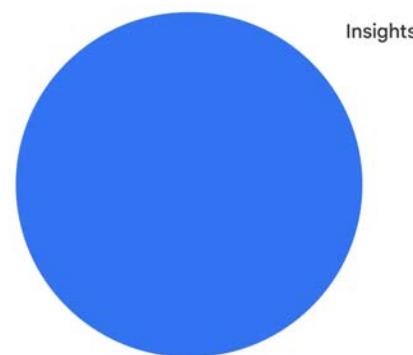
Google realized these infrastructure limitations and built solutions that:

1. **Scale with your data growth.**
2. **Are managed**, so you don't have to worry about infrastructure complexities.
3. **Allow you to focus on insights**, not system maintenance.

The key takeaway is that instead of building and maintaining your own infrastructure, you can leverage Google's global, scalable infrastructure.

## Google Cloud Network Infrastructure

**Big data with Google: Focus on insights, not infrastructure**

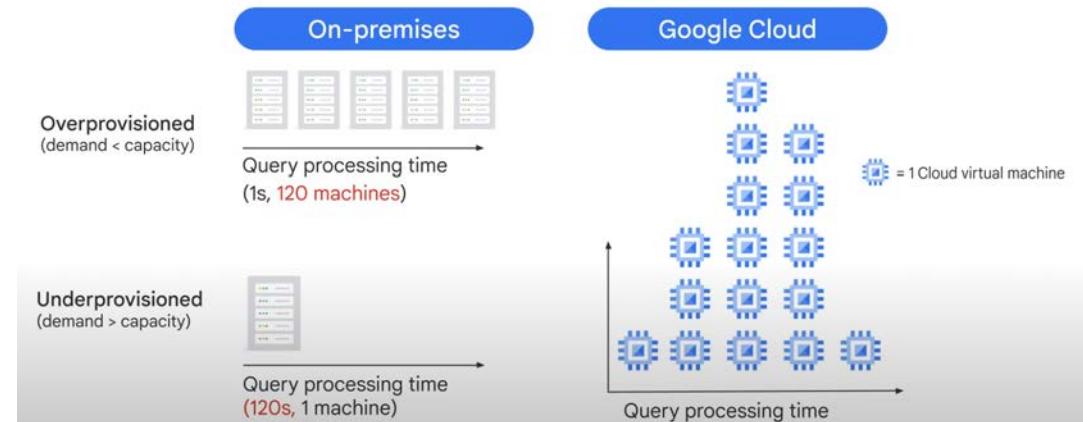


Google Cloud's network infrastructure is global and continually expanding. You can view the latest map of regions and zones at [Google Cloud Locations](#). The network's low-latency performance and scalability set it apart from other cloud service providers.

### The Benefits of Scalability

One significant benefit of cloud-based systems like Google Cloud is scalability:

#### Google Cloud enables on-demand scalability

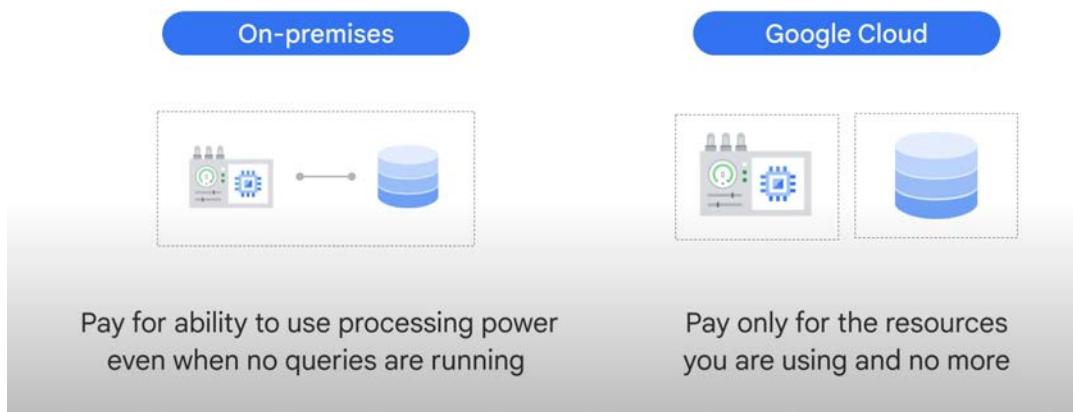


- On-premises systems require you to pay for power and infrastructure, even when it's not being used.
- In contrast, Google Cloud lets you scale up and down based on demand, ensuring that you only pay for the resources you use. For instance, you can

scale from 1 machine to 120 machines in seconds to handle large queries.

## Separation of Compute and Storage

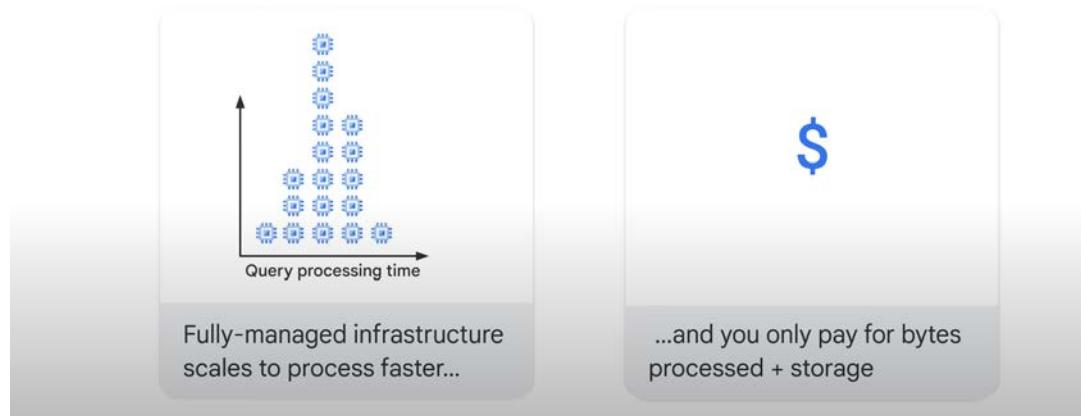
**Separation of storage and computing power enables efficient resource allocation**



A key feature of Google Cloud is the **decoupling of storage and compute power**. In on-premises setups, storage and compute resources are often tied together, meaning you pay for processing even when it's not in use. With Google Cloud, compute and storage are separated, so you pay only for the resources you actively use, such as storage or compute time.

## BigQuery for Data Analysts

**BigQuery scales automatically and you only pay for what you use**



BigQuery brings all these features together to make data analysis simpler and more efficient. It is Google Cloud's fully managed enterprise data warehouse designed to manage and analyze vast amounts of data. BigQuery's serverless architecture allows you to write SQL queries without worrying about infrastructure management.

- **Scalable Analysis:** BigQuery's distributed analysis engine lets you query terabytes of data in seconds and petabytes in minutes.
- **Flexible Storage and Compute:** By separating the compute engine from your storage, BigQuery maximizes flexibility and efficiency.

In summary, BigQuery helps you, the data analyst, focus on deriving insights from data, rather than spending time managing resources. Its scalability, flexibility, and powerful analysis tools make it an essential platform for modern data analysis.

## ▼ From Data to Insights with BigQuery

In this section, we will highlight the five common tasks of any data analyst and discuss how **BigQuery** can help with these tasks.

### 1. Understanding the Role of a Data Analyst

Previously, we briefly touched on the role of a data analyst and compared it with other data roles such as data engineers and data scientists.

### 2. Focus on Analysis and Visualization

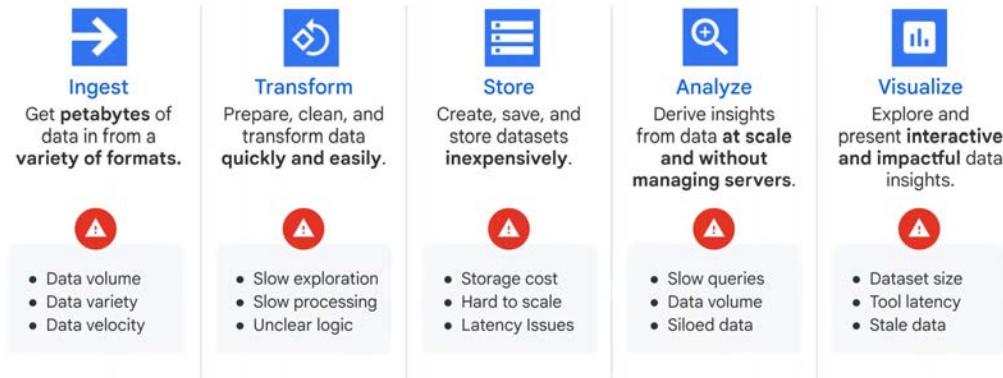
A data analyst is responsible for analyzing and gaining insights from data



Ideally, a data analyst should spend more time analyzing and visualizing data. We will explore these tasks in detail throughout this section. However, data analysts may also need to manage data ingestion, clean and transform data before analysis, and store it in an optimized way for faster retrieval.

### 3. Challenges Faced by Data Analysts

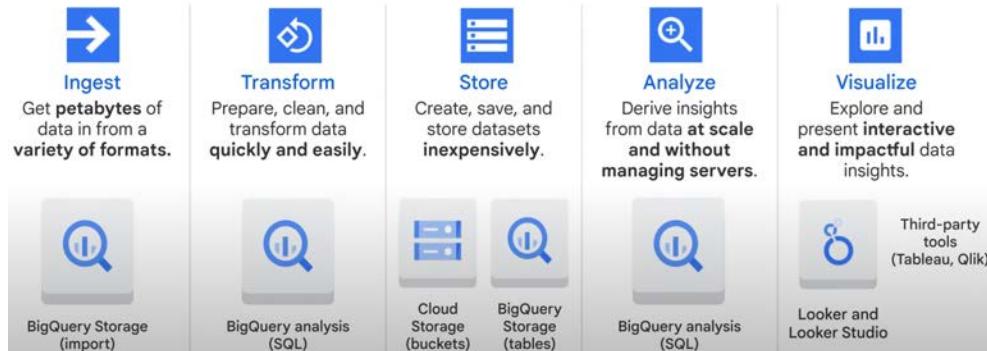
## Challenges in each task prevent data analysts from getting to scalable insights



Each of these tasks presents its own challenges. For example, analysis may slow down due to large data volumes or complex transformations, making it harder to get actionable insights. In this course, we will see how BigQuery helps overcome these challenges.

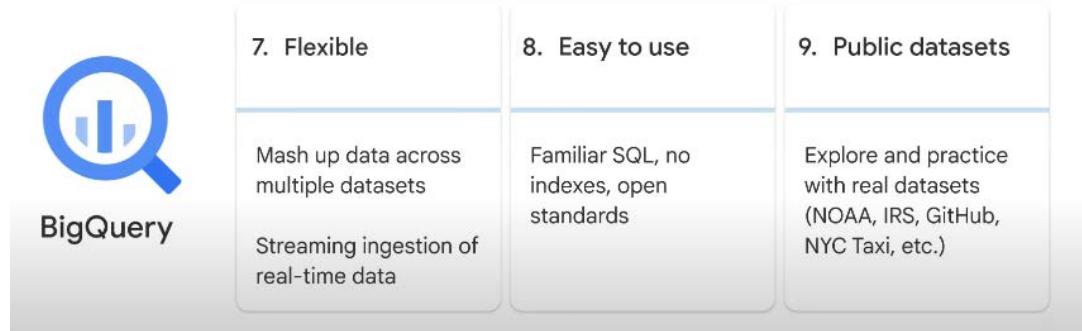
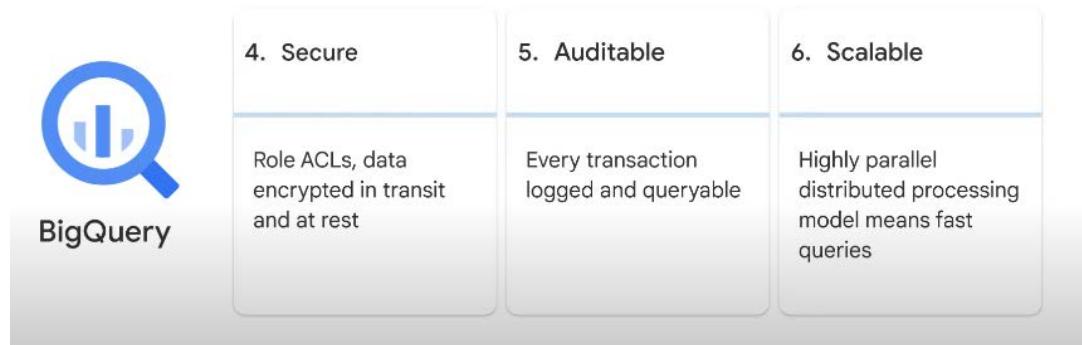
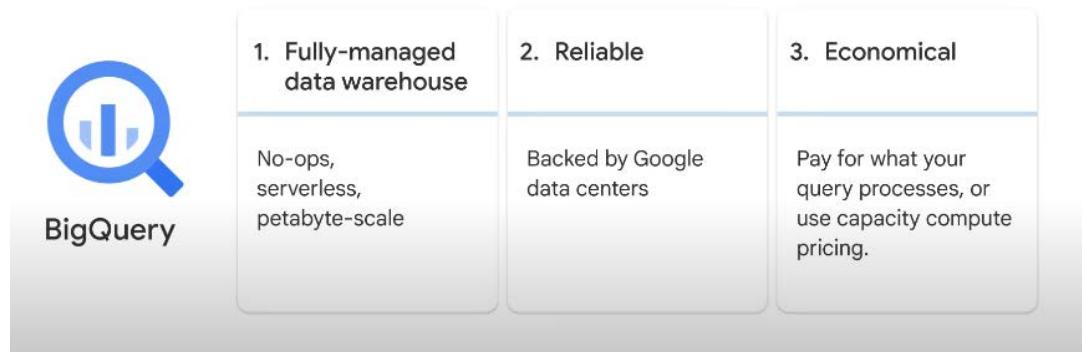
### 4. BigQuery's Key Features

#### BigQuery offers scalable tools to overcome data challenges



We will now discuss the main features of BigQuery, which make it a powerful solution for analytical workloads. These features provide scalability, cost efficiency, and performance improvements.

### BigQuery Features



- **Fully Managed Infrastructure**

BigQuery leverages Google's data center-backed infrastructure, meaning there's no need to worry about hardware optimization. Data analysts can focus on asking the right questions and uncovering insights from the data.

- **Enterprise Data Warehouse**

BigQuery offers a fully managed enterprise data warehouse, providing near real-time analysis of massive datasets. It runs on Google's secure and high-performance infrastructure, and its "NoOps" model means no administrative overhead for performance and scaling.

- **Secure and Reliable**

BigQuery data is replicated across multiple data centers, providing reliability. Additionally, data is encrypted both in transit and at rest. Access controls (ACLs) and Identity and Access Management (IAM) provide robust security measures, and Google Cloud Audit Logs track user activity.

- **Scalability and Speed**

BigQuery's virtually unlimited storage and processing power ensure fast query execution. Its highly parallel and distributed model enables efficient handling of large data volumes.

- **Real-time Data Streaming**

BigQuery allows for real-time data ingestion, making data available for analysis faster, without waiting for the complete dataset.

- **Flexibility with Diverse Datasets**

BigQuery supports mashing up data across various datasets and projects, including public datasets made available by BigQuery.

## BigQuery's Usability Features

- **Denormalized Tables and Columnar Storage**

Data is stored in simple, denormalized tables using columnar storage for high performance. No indexes, keys, or partitions are needed.

- **Familiar SQL Interface**

BigQuery uses a familiar SQL interface, intuitive UI, and supports nested and repeated fields for schema flexibility. Analysts can also use their preferred tools as BigQuery supports open standards.

- **Four Interaction Methods**

Analysts can interact with BigQuery using:

01

Web UI

Build, validate, and run queries quickly through the Web UI.

*This will be our primary focus for this course.*

02

Command-line interface (CLI)

Use Cloud Shell or the Google Cloud SDK (gcloud) to interact through a terminal.

```
bq mk [DATASET_ID]
```

03

REST API

Programmatically run queries using languages like Java and Python over HTTP.

GET

```
https://www.googleapis.com/bigquery/v2/projects/projectId/queries/jobId
```

04

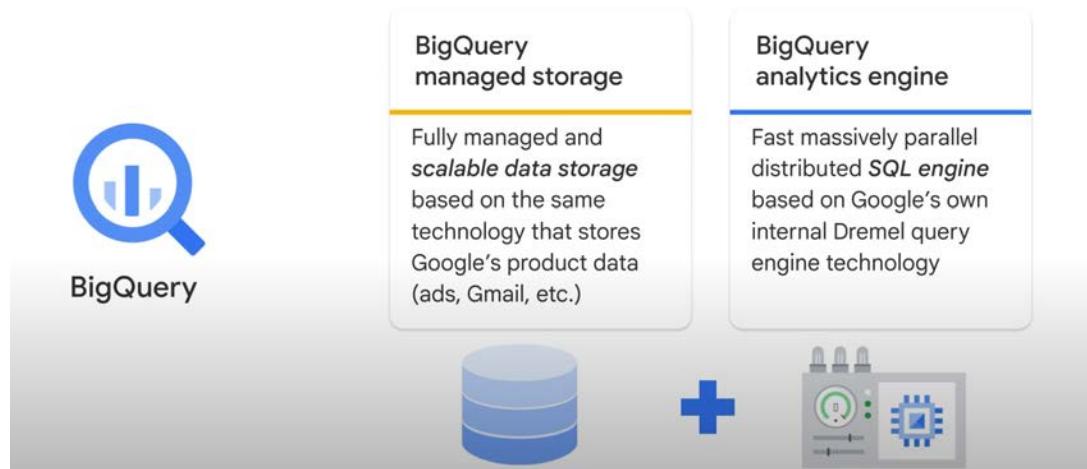
GUI SQL editors

Examples include Looker, Appsmith, Retool, Cogniti, superQuery, PopSQL, Zing Data, and many more.

## BigQuery's Dual Capabilities

BigQuery is a two-in-one service. It provides:

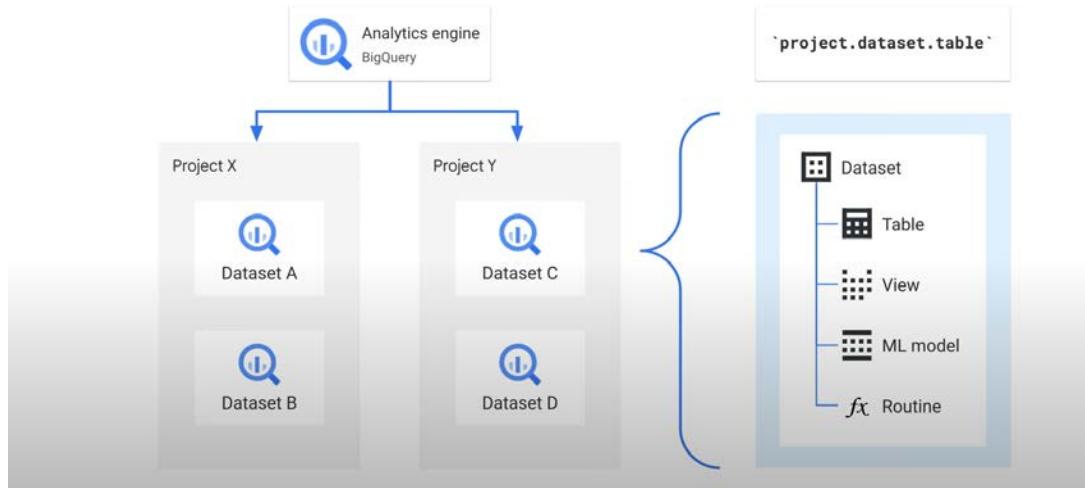
### BigQuery is actually two services in one



1. A **storage layer** for efficient and economical data storage, supporting multiple formats and data sources.
2. An **analysis engine** optimized for running queries on large datasets.

BigQuery also supports querying external data sources, which we will discuss in a later module.

## How BigQuery's resources are organized



Finally, remember that **tables** in BigQuery are organized within datasets, and datasets reside within projects. Datasets can also include other resources like views and routines, which will be covered later.

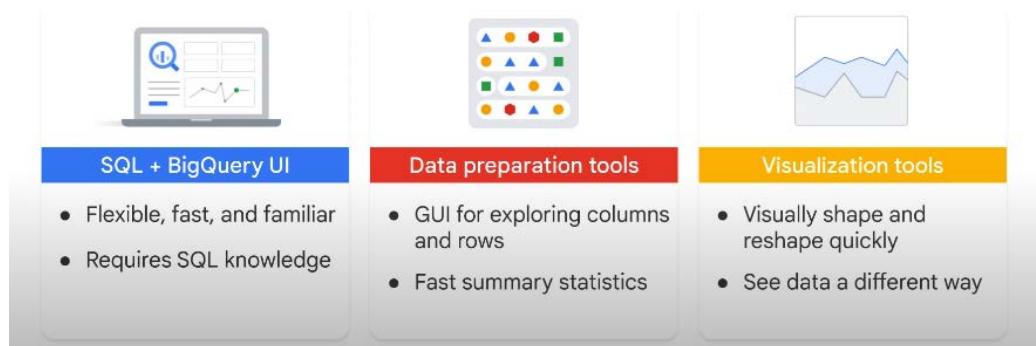
## ▼ 2- Exploring and Preparing Your Data with BigQuery

### ▼ Common data exploration techniques

In this lesson, we will discuss the various options available for exploring your datasets.

#### 1. Primary Options for Dataset Exploration

There are three main ways to explore a dataset:



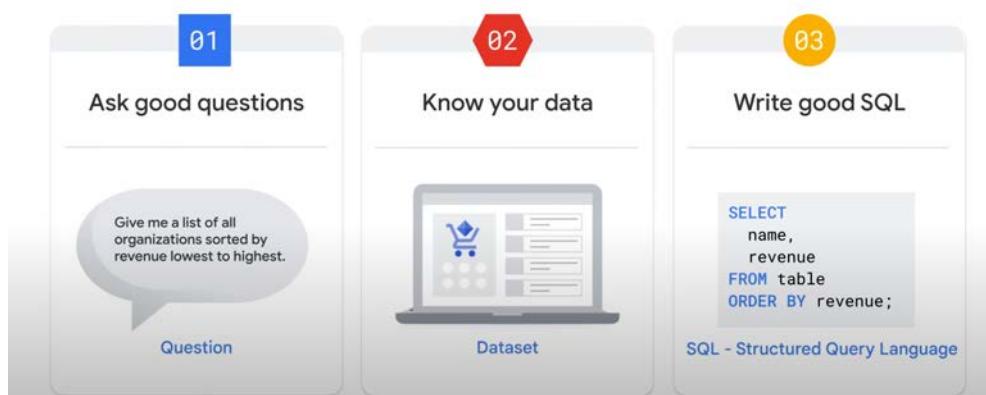
- **Writing SQL Queries:** You can write SQL queries and execute them in SQL editors like the BigQuery UI.

- **Data Preparation or BI Tools:** Tools such as Dataprep, SQL Runner in Looker, Looker Studio, and Cloud Data Fusion's Wrangler tool are great examples.
- **Visualization Tools:** You can also visualize raw data using various third-party tools.

## 2. Why SQL is Core for Data Analysts

- **SQL Syntax:** SQL has remained relatively unchanged since the 1980s, making it one of the most fundamental skills for a data analyst.
- **Beyond SQL:** Even though SQL is essential, as a data analyst, you are not restricted to using just SQL or the BigQuery Web UI. There are also data preparation and visualization tools, which we will briefly cover later.
- **Focus on SQL:** In this module, however, we will concentrate on SQL in the BigQuery Web UI because:
  - SQL is a crucial and fast way to interact with data in BigQuery.
  - It's also enjoyable for many analysts.

### Steps to explore data through SQL



## 1. Approaching Dataset Exploration

Exploring a dataset via SQL requires more than writing good queries. You need to understand your data's structure before writing a single line of code. Here are some guiding questions for exploration:

- **Data Interest:** What type of data am I interested in (e.g., financial data, nonprofit organizations)?
- **Specific Insights:** What specific information am I looking for (e.g., organization revenue)?

- **Result Formatting:** How do I want the results presented (e.g., sorted by highest revenue)?

## 2. Dataset Understanding

Before querying, consider the following:

- **Dataset Availability:** What datasets are available to me? Do I need to upload my own data?
- **Data Structure:** How is the data structured? Are there multiple tables? What are the important fields?
- **Data Size:** How much data do I need to explore?

## 3. Formulating SQL Queries

When writing SQL queries, think about:

- How do I translate my question into an SQL query?
- Is my data clean?
- What fields should I focus on?
- Do I need to perform any aggregations?
- Do I need data from multiple tables?

These steps are essential for writing efficient queries that help you explore data quickly and gather the insights you need.

In the remainder of this module, we will go through each of these steps in detail and write queries to extract valuable insights from our datasets.

## ▼ Analyze large datasets with BigQuery

In this lesson, we will briefly discuss the datasets that we will use to run SQL queries.

### 1. Public Datasets in BigQuery

Public datasets include flights, taxi cab logs, weather recordings, and many more.

Example SQL code is provided for practice.



- A **public dataset** is any dataset stored in BigQuery that is made available to the general public through the Google Cloud Public Dataset Program.
- These datasets are hosted by BigQuery, allowing you to access and integrate them into your applications.
- Throughout this course, we will use several public datasets to demonstrate how to build and run SQL queries in the BigQuery UI.

## 2. Real-World Ecommerce Dataset

- One of the challenges in building a course like this is selecting a practice dataset that analysts can relate to and query for meaningful insights.
- In this course, we are excited to present a **real ecommerce dataset** with over a million site hits, containing a year of transaction records from **Google's online store** (where Google-branded merchandise is sold).
- These transactions are recorded through **Google Analytics** and made publicly available via BigQuery.

## 3. Practical Ecommerce Questions

With the SQL and BigQuery skills you'll gain in this course, you'll be able to answer real-world ecommerce questions, such as:

- What is the total number of transactions generated per device browser?
- Which customers have added items to their carts but then abandoned them?
- How can I create cohorts or segments based on customer behavior for more personalized targeting?
- What keywords or referring partner sites led visitors to the ecommerce site, and what pages did they visit?

## 4. Real-World Applicability

- The best part of working with this dataset is that the queries and insights you derive will be **directly applicable** to your own Google Analytics datasets.
- Even if you're not focused on ecommerce, the **analytical techniques** we cover will be useful for your own SQL queries in other contexts.

# ▼ Query basics

In most cases, queries in BigQuery follow the same principles as those used in other databases. There is an international standard for SQL, with each version being referred to by the term ANSI and the year it was adopted. BigQuery SQL

adheres to the ANSI 2011 standards. However, like many SQL-based databases, BigQuery has some minor platform-specific differences.

## Standard SQL Constructs

BigQuery supports the usual SQL constructs such as `SELECT`, `WHERE`, `JOIN`, `GROUP BY`, and `ORDER BY`. You'll also find common aggregation functions like `COUNT`, `SUM`, `MIN`, and `MAX`, which are standard in most SQL implementations.

## BigQuery-Specific Features

Some functionality has been added to support features unique to BigQuery, such as:

- Arrays and structs
- Geography data types
- BigQuery Machine Learning (BigQuery ML)

## SQL Query Basics

A typical SQL query begins with the `SELECT` keyword, followed by one or more columns you want in your results. For example, if you want to view the `name` and `revenue` columns, you would specify those in your `SELECT` statement.

```
SELECT  
    name,  
    revenue  
FROM sales_table;
```

You must specify the table using the `FROM` clause, and you can filter rows using the `WHERE` clause. For example, to find products with revenue greater than \$100,000:

```
SELECT
    name,
    revenue
FROM sales_table
WHERE revenue > 100000;
```

Filters can be combined, like so:

```
SELECT
    name,
    revenue
FROM sales_table
WHERE revenue > 100000 AND category = 'Toys';
```

## Sorting Results

You can sort results using the `ORDER BY` clause, which defaults to ascending order:

```
SELECT
    name,
    revenue
FROM sales_table
ORDER BY revenue;
```

For descending order, use `DESC`:

```
SELECT
    name,
    revenue
FROM sales_table
WHERE revenue > 100000
ORDER BY revenue DESC, category;
```

## Limiting Results

If you only need a subset of rows, use the `LIMIT` clause:

```
SELECT
    name,
    revenue
FROM sales_table
LIMIT 10;
```

## Commenting Queries

It's good practice to comment your SQL queries, especially as they become more complex. Comments help explain the logic of your queries.

```
# limit to only 10 rows
-- another way to do single line comment
SELECT
    name,
    revenue
FROM sales_table
LIMIT 10;
```

```
/* Use this if
your comment is
long,
or spans
multiple lines.
*/
```

Note: # is specific to BigQuery, while -- is SQL ANSI standard, which is supported in BigQuery and most other SQL databases.

## BigQuery Table Structure

In BigQuery, tables reside within datasets, and datasets exist within projects. When referencing tables, it's a good idea to use fully qualified names in the format:

`project_id.dataset.table`.

```
SELECT  
    fullVisitorId,  
    country,  
    timeOnSite  
FROM all_sessions  
LIMIT 10;
```

Remember that tables in BigQuery exist within a dataset.

```
SELECT  
    column  
FROM dataset.table;
```

So it's important to specify the dataset name. Tables exist in datasets, and datasets exist within projects. And if you omit the project ID, BigQuery will assume the project is your current one.

This could be an issue when, for example, you are trying to access public datasets, as they would be hosted in the `bigrquery-public-data` project.

```
SELECT  
    column  
FROM `project-id.dataset.table`;
```

Use back ticks (`) to get a suggestions list as you type out the fully qualified name of the table.

So it's a good idea to use the fully qualified name using the notation `'project-id.dataset.table_name'`. Use back ticks (`) to get a suggestions list as you type out the fully qualified name of the table, starting with the project ID.

## Using Wildcards and Previewing Data

While using the `*` wildcard selects all columns, be cautious, especially with large tables, as this can incur significant costs. For exploration, use the `Preview` tab in BigQuery to inspect the schema and some data without running a costly query.

```
SELECT *
FROM `data-to-insights.ecommerce.all_sessions`;
```

If the intention is to explore the schema of a new table, you can use the Schema tab. Running `SELECT *` without a `LIMIT` or `WHERE` clause on a large table will cost a lot as the entire table, all rows, all fields, are read and returned.

## Query Costs and Caching

BigQuery charges based on the number of bytes processed. You get 1 TB of query processing per month at no cost. Query results are cached for approximately 24 hours, and if you re-run the exact same query, cached results will be used without additional charges.

## Combining ORDER BY and LIMIT

Now we did see examples of using using the `ORDER BY` and `LIMIT` clauses in previous SQL queries. Combining them in the same query helps you when you need to do “top n” analysis.... like for example the top 10 products by revenue, or top 5 most popular patents, etc.

```

SELECT
    fullVisitorId,
    country,
    timeOnSite
FROM `data-to-insights.ecommerce.all_sessions`
ORDER BY timeOnSite DESC
LIMIT 10;

```

In this example, you are ordering the rows by the timeOnSite column (which shows the amount of time the website visitors spent on the site).

| Row | fullVisitorId       | country       | timeOnSite |
|-----|---------------------|---------------|------------|
| 1   | 0824839726118485274 | United States | 19017      |
| 2   | 0824839726118485274 | United States | 19017      |
| 3   | 0824839726118485274 | United States | 19017      |
| 4   | 0824839726118485274 | United States | 19017      |
| 5   | 0824839726118485274 | United States | 19017      |
| 6   | 0824839726118485274 | United States | 19017      |
| 7   | 0824839726118485274 | United States | 19017      |
| 8   | 0824839726118485274 | United States | 19017      |
| 9   | 0824839726118485274 | United States | 19017      |
| 10  | 0824839726118485274 | United States | 19017      |

| Row | fullVisitorId       | country       | timeOnSite |
|-----|---------------------|---------------|------------|
| 1   | 0824839726118485274 | United States | 19017      |
| 2   | 2706961341001088633 | United States | 15047      |
| 3   | 9894955795481014038 | Venezuela     | 15020      |
| 4   | 596895434219823695  | Venezuela     | 14279      |
| 5   | 4742180546650265795 | Panama        | 12853      |
| 6   | 9264804092676520813 | Venezuela     | 12466      |
| 7   | 6957245643416321514 | United States | 12136      |
| 8   | 1957458976293878100 | United States | 11848      |
| 9   | 8826538902252293768 | United States | 11316      |
| 10  | 7498695963354635199 | United States | 11275      |

And if your resulting rows have duplicates, like what you see in the result on the left here, then use **SELECT DISTINCT** to remove duplicates.

Another thing you may have noticed is that the timeOnSite column shows durations in seconds. So you can use a simple mathematical formula to change that into minutes.

```

SELECT DISTINCT
    fullVisitorId,
    country,
    timeOnSite / 60 AS session_time_minutes
FROM
`data-to-insights.ecommerce.all_sessions`
ORDER BY session_time_minutes DESC
LIMIT 10;

```

| Row | fullVisitorId       | country       | session_time_minutes |
|-----|---------------------|---------------|----------------------|
| 1   | 0824839726118485274 | United States | 316.95               |
| 2   | 2706961341001088633 | United States | 250.7833333333333    |
| 3   | 9894955795481014038 | Venezuela     | 250.3333333333334    |
| 4   | 596895434219823695  | Venezuela     | 237.9833333333332    |
| 5   | 4742180546650265795 | Panama        | 214.216666666666667  |
| 6   | 9264804092676520813 | Venezuela     | 207.766666666666668  |
| 7   | 6957245643416321514 | United States | 202.266666666666668  |
| 8   | 1957458976293878100 | United States | 197.466666666666667  |
| 9   | 8826538902252293768 | United States | 188.6                |
| 10  | 7498695963354635199 | United States | 187.916666666666666  |

How can we clean that up a little?

```

SELECT DISTINCT
    fullVisitorId,
    country,
    ROUND(timeOnSite / 60,2) AS session_time_minutes
FROM
`data-to-insights.ecommerce.all_sessions`
ORDER BY session_time_minutes DESC
LIMIT 10;

```

| Row | fullVisitorId       | country       | session_time_minutes |
|-----|---------------------|---------------|----------------------|
| 1   | 0824839726118485274 | United States | 316.95               |
| 2   | 2706961341001088633 | United States | 250.78               |
| 3   | 9894955795481014038 | Venezuela     | 250.33               |
| 4   | 596895434219823695  | Venezuela     | 237.98               |
| 5   | 4742180546650265795 | Panama        | 214.22               |
| 6   | 9264804092676520813 | Venezuela     | 207.77               |
| 7   | 6957245643416321514 | United States | 202.27               |
| 8   | 1957458976293878100 | United States | 197.47               |
| 9   | 8826538902252293768 | United States | 188.6                |
| 10  | 7498695963354635199 | United States | 187.92               |

You can use the `ROUND` function that will round up the value to 2 decimal places.

## Functions in SQL Queries

BigQuery supports a wide range of functions, such as `ROUND`, which allows you to round values to a specified number of decimal places:

```

ROUND(<field>, <decimals>)
Function = Performs an Action
Parameters = Inputs you provide

```

```
SELECT DISTINCT
    fullVisitorId,
    country,
    ROUND(timeOnSite / 60,2) AS
    session_time_minutes
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE session_time_minutes > 240
ORDER BY session_time_minutes DESC
LIMIT 10;
```



Unrecognized name:  
session\_time\_minutes

Well, newly defined aliases in the SELECT statement cannot be used yet for filtering rows in your WHERE clause.

And the reason for that is that when the query reads the table from disk, it filters for the columns you want returned (which is basically your WHERE clause).

```
SELECT DISTINCT
    fullVisitorId,
    country,
    ROUND(timeOnSite / 60,2) AS
    session_time_minutes
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE ROUND(timeOnSite / 60,2) > 240
ORDER BY session_time_minutes DESC
LIMIT 10;
```

| Row | fullVisitorId       | country       | session_time_minutes |
|-----|---------------------|---------------|----------------------|
| 1   | 0824839726118485274 | United States | 316.95               |
| 2   | 2706961341001088633 | United States | 250.78               |
| 3   | 9894955795481014038 | Venezuela     | 250.33               |

## Hotkeys and Additional Resources

For further exploration, you can refer to demos such as "explore-data-with-sql.sql" available on GitHub. Use hotkeys in the BigQuery interface for an efficient workflow.

- [Demo using hotkeys](#)

This guide provides an overview of query basics and tips to optimize your use of BigQuery for querying and data analysis.

## ▼ Working with functions

In this section, we'll explore various functions and how they can be utilized within SQL queries in BigQuery.

### String Manipulation Functions

String functions allow you to perform transformations on string data, such as altering its format. These functions can modify strings in a variety of ways, like

converting text to uppercase or extracting specific characters from a string.

For example:

- The **CONCAT** function joins two or more strings or byte values into a single result. You can add a format function to format the results in a certain way, for example displaying revenue column values with commas, so it reads much better.

- `CONCAT("12345", "678")`
  - "12345678"

- Here are a few common string functions: The CONCAT function concatenates 2 or more strings together.

- **LOWER** converts all characters in a string to lowercase.

- `LOWER("Apple")`
  - "apple"

- **ENDS\_WITH** checks if the second string is a suffix of the first string, returning a boolean value.

- `ENDS_WITH("Apple", "e")`
  - true

- The last function here, **REGEXP\_CONTAINS**, returns true if the value is a partial match to the regular expression.

- `REGEXP_CONTAINS("Lunchbox", r"^\*box$")`
  - true

- This example uses a string function on the name, to render the product name in lowercase, and then finds the word 'shirt' anywhere in the name.

```

SELECT DISTINCT
    v2ProductName
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE LOWER(v2ProductName) LIKE '%shirt%'
LIMIT 100;

```

| Row | v2ProductName                                 |
|-----|---|
| 1   | BLM Sweatshirt                                |
| 2   | Google Toddler Raglan Shirt Blue Heather/Navy |
| 3   | BLM Sweatshirt (Pre-Order)                    |
| 4   | Google Toddler Short Sleeve T-shirt Green     |
| 5   | Google Women's Vintage T-Shirt Black          |
| 6   | Google Youth Short Sleeve T-shirt Green       |
| 7   | Android Toddler Short Sleeve T-shirt Pink     |
| 8   | Android Youth Short Sleeve T-shirt Pewter     |

When dealing with string matching, the **LIKE** operator is often used to perform basic pattern matching. For instance, searching for the word "shirt" in a product name can be done using `LIKE '%shirt%'`. However, it's important to use functions like **LOWER** to ensure consistency when comparing values, especially if the case format of the data is unknown.

Note that using **LOWER** or other transformations on large datasets can be costly, as BigQuery needs to read and process all the values before performing comparisons.

## Aggregation Functions

Aggregation functions are essential for summarizing data. Common ones include:

- **COUNT**: Counts the number of values.
  - You can see the COUNT function here giving the total number of unique users as it aggregates the total number of distinct visitor IDs. This kind of information is great when looking at website traffic.

```

SELECT
    COUNT(DISTINCT fullVisitorId) AS unique_users
FROM
    `data-to-insights.ecommerce.all_sessions`;

```

| Row | unique_users |
|-----|--------------|
| 1   | 389934       |

- Can we do more, though?

We know that there is country information captured in the table as well.

Can we do a total count by country?

The answer is yes, and this is where you get to use the GROUP BY clause. So essentially, you are getting the aggregate total, but after grouping the visitors based on a certain column, which in this case, is the country. Now the output is more exciting and insightful as we can see the top 5 countries, in terms of unique visitors to the website.

```

SELECT
    COUNT(DISTINCT fullVisitorId) AS unique_users
FROM
    `data-to-insights.ecommerce.all_sessions`;

```

| Row | unique_users |
|-----|--------------|
| 1   | 389934       |

- Filter duplicates with COUNT and HAVING
  - Sometimes we want to filter results after some sort of aggregation. The way to do it is using the HAVING clause.
  - In this example, we want to look at visits per website visitor, but only for those who have visited more than once. And so we use the total count of visits, which is aliased as the records column, in the HAVING clause.
  - You can think of the HAVING clause as the WHERE clause for aggregated columns.

```

SELECT
    fullVisitorId,
    COUNT(fullVisitorId) AS records
FROM
    `data-to-insights.ecommerce.all_sessions`
GROUP BY fullVisitorId
HAVING records > 1
LIMIT 10;

```

| Row | fullVisitorId       | records |
|-----|---------------------|---------|
| 1   | 8919336618754256169 | 1146    |
| 2   | 4605997863482509872 | 276     |
| 3   | 7228946486690765003 | 264     |
| 4   | 912127983342148918  | 151     |
| 5   | 7419028556980464579 | 68      |
| 6   | 832152661091318994  | 105     |
| 7   | 192907985600193802  | 504     |
| 8   | 0267830135658533597 | 278     |
| 9   | 0794270160070827977 | 171     |
| 10  | 6949496197162068722 | 556     |

- **SUM:** Adds up values.
- **MIN:** Finds the minimum value.
- **MAX:** Finds the maximum value.

When performing aggregations, the **GROUP BY** clause is often used. For example, counting unique website visitors by country involves grouping the visitor data by the country column. Additionally, you can filter the results after aggregation using the **HAVING** clause, such as showing only visitors with more than one visit.

## Data Type Conversion with CAST

You can convert data types using the **CAST** function. This is useful when you need to treat a column as a different type.

```
CAST(expression AS typename [format_clause])
```

For example:

- `SELECT CAST("12345" AS INT64)`
  - 12345
- `SELECT CAST("2017-08-01" AS DATE)`
  - 2017-08-01
- `SELECT CAST(1112223333 AS STRING)`
  - "1112223333"
- `SELECT SAFE_CAST("apple" AS INT64)`
  - NULL

- Now sometimes, `CAST` may not work because you are converting a string column into an integer, and maybe some of the values in the column are not valid.
- If you tried the `CAST` function normally, this would error out. But you can use **SAFE CAST** instead.
- Now it just produces a **NULL** for that value instead of throwing an error.

## What is NULL value?

Well, just like black holes are the absence of light, NULLs are the absence of data.

| Row | blank_field | null_field |
|-----|-------------|------------|
| 1   |             | null       |

```
[  
  {  
    "blank_field": "",  
    "null_field": null  
  }  
]
```

- NULLs are valid values
- NULL is the absence of data or an empty set
- NULL is not the same as "" or a valid blank string value
- NULL != NULL

- NULLs may or may not be included in aggregations.
- By default NULLs are respected.
- You cannot simply do IF VALUE = NULL since NULLS can never be equivalent to anything, Null is not even equal to Null.
- So if you want to filter out null values, you can specify IS NOT NULL in the WHERE clause.
- This example will display only rows that have a valid transaction ID, and discard the ones where the transaction ID value is null.

```
SELECT DISTINCT
    fullVisitorId,
    date,
    time,
    pageviews,
    pageTitle,
    v2ProductName,
    transactionId
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE transactionId IS NOT NULL
ORDER BY date, time
LIMIT 100
```

## Date Function

- Converting a string into a date so it can be used with functions like **EXTRACT** to retrieve parts of the date (e.g., the month).

```
EXTRACT(part FROM date_expression)
```

- For example

```
SELECT EXTRACT(DAY FROM DATE '2013-12-25') AS the_day;

/* -----
   | the_day |
   +-----+
   | 25      |
   *-----*/
```

## BigQuery Data Types

| 1.9  | ABC   | 2016  | 1/0   |
|--|---|---|---|
| <b>Numeric data</b><br>Integer (int64)<br>Whole numbers that can be negative<br>(-2,-1,0,1,2,3,4)<br><br>Float (float64)<br>(1.0000000001) | <b>String data</b><br>Strings<br>Text values<br>(‘dog’, ‘cat’, ‘800-999-9999’)<br><br>In SQL, use single quotes when dealing with strings like ‘Google LLC’ | <b>Dates</b><br>Dates (datetime)<br>Stored in universal time format. Allowable range: 0001-01-01 00:00:00 to 9999-12-31 23:59:59.999999 | <b>Other</b><br>Boolean (TRUE/FALSE)<br>Array [‘apple’, ‘pear’]<br>Struct<apple string> |
|  |   |   |   |

We have seen most of them in the queries and results we have discussed so far. Take note that this is not the complete list.

To learn more, you can go to:

<https://cloud.google.com/bigquery/docs/reference/standard-sql/data-types>

## Aggregating Multiple Rows

If you have multiple rows with similar data, you can use aggregation functions to combine information.

| Row | fullVisitorId       | date     | time   | pageviews | pageTitle             | v2ProductName  | transactionId  |
|-----|---------------------|----------|--------|-----------|-----------------------|--|----------------|
| 1   | 4631129802514106099 | 20160801 | 144555 | 15        | Checkout Confirmation | 24 oz YouTube Sergeant Stripe Bottle                 | ORD20160801423 |
| 2   | 6027268712782791947 | 20160801 | 210118 | 21        | Checkout Confirmation | Google Men's 100% Cotton Short Sleeve Hero Tee White | ORD20160801430 |
| 3   | 6027268712782791947 | 20160801 | 210118 | 21        | Checkout Confirmation | Google Men's 100% Cotton Short Sleeve Hero Tee Red   | ORD20160801430 |
| 4   | 5563168194966233133 | 20160801 | 259524 | 17        | Checkout Confirmation | Google Infant Zip Hood Pink                          | ORD20160801436 |
| 5   | 5563168194966233133 | 20160801 | 259524 | 17        | Checkout Confirmation | Google Baby Essentials Set                           | ORD20160801436 |
| 6   | 1468560120795000800 | 20160801 | 282300 | 17        | Checkout Confirmation | Deluge Waterproof Backpack                           | ORD20160801443 |
| 7   | 1468560120795000800 | 20160801 | 282300 | 17        | Checkout Confirmation | PaperMate Ink Joy Retractable Pen                    | ORD20160801443 |
| 8   | 1468560120795000800 | 20160801 | 282300 | 17        | Checkout Confirmation | Google Men's 100% Cotton Short Sleeve Hero Tee Black | ORD20160801443 |

- For example, the **STRING\_AGG** function concatenates string values from different rows into a single string.

```

SELECT
    transactionId,
    (totalTransactionRevenue / 1000000) AS revenue,
    STRING_AGG(v2ProductName) AS product_list
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE transactionId IS NOT NULL
GROUP BY transactionId, totalTransactionRevenue
ORDER BY revenue DESC
LIMIT 100

```

- And so here you see one row per transaction ID (because it was one of the grouping columns), and the list of products all combined together into a single string.

| Row | transactionId   | revenue  | product_list   |
|-----|-----------------|----------|--|
| 1   | ORD201704052324 | 47082.06 | Leatherette Journal,Google 5-Panel Cap,Google Luggage Tag,YouTube Twill Cap,Google Sunglasses,Red Spiral Google Notebook,20 oz Stair   |
| 2   | ORD201704182260 | 32153.82 | Android Baby Essentials Set,Android Youth Short Sleeve T-shirt Aqua,Google Women's Short Sleeve Hero Tee Sky Blue,Android Lifted Men's Short Sleeve Tee Blue,YouTube Youth Short Sleeve Tee Red,Google Youth Short Sleeve Tee Red,Google Toddler Short Sleeve Tee White,Android Toddler Short Sleeve T-shirt Pink,Android Youth Short Sleeve T-shirt Pewter,YouTube Youth Short Sleeve Tee Red,Android Toddler Short Sleeve T-shirt Aqua,Android Lifted Men's Short Sleeve Tee Blue,Android Baby Essentials Set,Android Toddler Short Sleeve T-shirt Pink,Google Youth Short Sleeve T-shirt Royal Blue,Google Youth Short Sleeve T-shirt Royal Blue,YouTube Youth Short Sleeve Tee Red,Google Men's Zip Hoodie,Google Men's Zip Hoodie,Google Toddler Short Sleeve T-shirt Royal Blue,Android Toddler Short Sleeve T-shirt Pink,Google Youth Short Sleeve Tee Red,Google Spiral Journal with Pe... |
| 3   | ORD201707182786 | 25251.26 | Google Luggage Tag,YouTube Twill Cap,Google Twill Cap,YouTube Hard Cover Journal,Sport Bag,Google Water Resistant Bluetooth Speaker  |
| 4   | ORD201702142258 | 17859.5  | Google Men's 100% Cotton Short Sleeve Hero Tee White,Google Wool Heather Cap Heather/Navy,Recycled Mouse Pad,Yoga Mat  |

- Similarly, **ARRAY\_AGG** aggregates values into an array, providing a more readable format when dealing with multiple products or categories within a single transaction.

```

SELECT
    transactionId,
    (totalTransactionRevenue / 1000000) AS revenue,
    ARRAY_AGG(v2ProductName) AS product_list
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE transactionId IS NOT NULL
GROUP BY transactionId, totalTransactionRevenue
ORDER BY revenue DESC
LIMIT 100

```

- The function works in the same way by combining all the products together, but presentation-wise it's a little different.

| Row | transactionId   | revenue  | product_list                            |
|-----|-----------------|----------|---|
| 1   | ORD201704052324 | 47082.06 | Google 22 oz Water Bottle               |
|     |                 |          | Colored Pencil Set                      |
|     |                 |          | Leatherette Journal                     |
|     |                 |          | Google Sunglasses                       |
|     |                 |          | Google Luggage Tag                      |
|     |                 |          | YouTube Twill Cap                       |
|     |                 |          | Android Luggage Tag                     |
|     |                 |          | YouTube Luggage Tag                     |
|     |                 |          | Google Sunglasses                       |
|     |                 |          | 20 oz Stainless Steel Insulated Tumbler |

As you can see here, instead of a really long string of product names mashed up together, here you see an array of product names, which is a lot more readable. The same grouping applies so each transaction ID is only displayed once.

- This example uses the array aggregation function on the product name and the product quantity, so you can see the amount bought per product, and a nice summary of the number of distinct products per transaction.

| Row | transactionId   | revenue  | distinct_products_ordered | product_quantity | product_list                            |
|-----|-----------------|----------|---------------------------|------------------|---|
| 1   | ORD201704052324 | 47082.06 | 15                        | 200              | Google 22 oz Water Bottle               |
|     |                 |          |                           | 400              | Colored Pencil Set                      |
|     |                 |          |                           | 100              | Google Luggage Tag                      |
|     |                 |          |                           | 144              | YouTube Twill Cap                       |
|     |                 |          |                           | 600              | Leatherette Journal                     |
|     |                 |          |                           | 500              | Google 5-Panel Cap                      |
|     |                 |          |                           | 750              | Google Sunglasses                       |
|     |                 |          |                           | 100              | Android Luggage Tag                     |
|     |                 |          |                           | 200              | 20 oz Stainless Steel Insulated Tumbler |
|     |                 |          |                           | 1000             | YouTube Custom Decals                   |

There are many other types of functions that have not yet been discussed.

## WITH Clause and Common Table Expressions (CTEs)

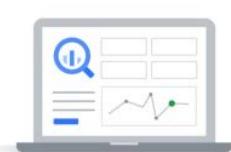
The **WITH** clause allows you to define temporary tables within a query, called Common Table Expressions (CTEs). This can be useful for breaking complex queries into smaller, more manageable parts. Each CTE binds the result of a subquery to a table name, which can then be referenced throughout the rest of the query.

```
WITH product_views AS (
    # all products
    SELECT
        COUNT(DISTINCT fullVisitorId) AS visitors,
        v2ProductName
    FROM
        `data-to-insights.ecommerce.all_sessions`
    GROUP BY v2ProductName
)
# popular shirts
SELECT * FROM product_views
WHERE LOWER(v2ProductName) LIKE '%shirt%'
AND visitors > 10000
ORDER BY visitors DESC
```

WITH clauses (common table expressions) are effectively sub-queries and can be changed together.

If you are continually using a certain WITH clause, consider promoting it to a view or table (covered later).

## Summary



Explore large datasets quickly with SQL in the BigQuery UI



Comment your code for better readability



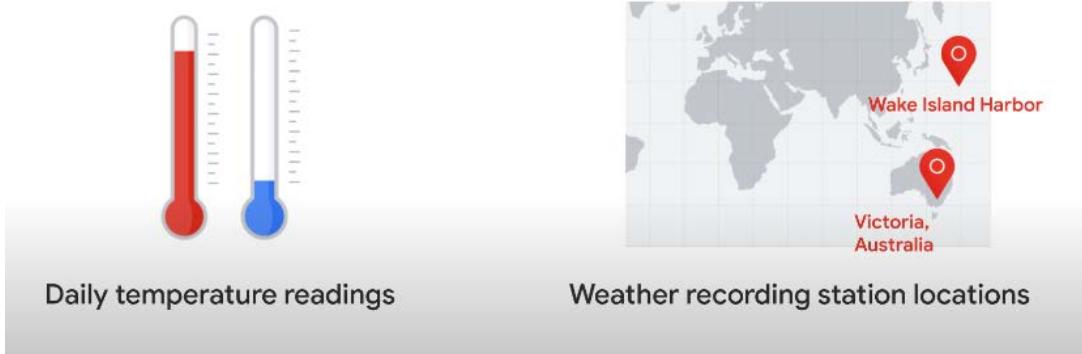
Fix common SQL syntax errors by using the validator



Practice SQL skills on hundreds of public datasets with sample queries

## ▼ Enrich Your Queries with Unions and Joins

In this section, we'll explore how to use **unions** and **joins** to enhance your queries. For our examples, we will use **temperature readings** and **weather station data** from BigQuery's NOAA public dataset. Understanding how to work with unions and joins allows you to combine data from multiple tables for enriched analysis.



## Data Description

The NOAA dataset contains two key types of data:

| Daily temperature readings |             | Weather station location details |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
|----------------------------|-------------|----------------------------------|-------------|-----------------|-----|--------|-------|----------------------|---------|-------|------|--------|--------|---------|----------|----------|
|                            | ... current | Results                          | Explanation | Job Information | Row | usaf   | wban  | name                 | country | state | call | lat    | lon    | elev    | begin    | end      |
| ▼ noaa_gsod                |             |                                  |             |                 | 1   | 912450 | 41806 | WAKE ISLAND AIRFLD   | WQ      | PC    | PWAK | 19.283 | 166.65 | +0003.7 | 19451231 | 20100731 |
| ■ gsod1929                 | ■ gsod1942  | ■ gsod1956                       |             |                 | 2   | 912460 | 41606 | WAKE ISLAND AIRFIELD | WQ      | UM    | PWAK | 19.283 | 166.65 | +0007.0 | 20100801 | 20170805 |
| ■ gsod1930                 | ■ gsod1943  | ■ gsod1957                       |             |                 | 3   | 999999 | 41606 | WAKE ISLAND          | WQ      | PC    | PWAK | 19.283 | 166.65 | +0003.7 | 19491031 | 19721231 |
| ■ gsod1931                 | ■ gsod1944  | ■ gsod1958                       |             |                 | 4   | 912450 | 99999 | WAKE ISLAND AIRFLD   | WQ      |       |      | 19.283 | 166.65 | +0004.0 | 20000101 | 20100818 |
| ■ gsod1932                 | ■ gsod1945  | ■ gsod1959                       |             |                 | 5   | 997387 | 99999 | WAKE ISLAND          | WQ      |       |      | 19.28  | 166.62 | +0005.0 | 20050517 | 20170804 |
| ■ gsod1933                 | ■ gsod1946  | ■ gsod1960                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1934                 | ■ gsod1947  | ■ gsod1961                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1935                 | ■ gsod1948  | ■ gsod1962                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1936                 | ■ gsod1949  | ■ gsod1963                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1937                 | ■ gsod1950  | ■ gsod1964                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1938                 | ■ gsod1951  | ■ gsod1965                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1939                 | ■ gsod1952  | ■ gsod1966                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1940                 | ■ gsod1953  | ■ gsod1967                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1941                 | ■ gsod1954  | ■ gsod1968                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1942                 | ■ gsod1955  | ■ gsod1969                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |
| ■ gsod1943                 | ■ gsod1956  | ■ gsod1970                       |             |                 |     |        |       |                      |         |       |      |        |        |         |          |          |

- Daily temperature readings:** A separate table exists for each year starting from 1929. For example, one table for 1929, another for 1930, and so on.
- Weather station details:** Information such as latitude, longitude, state, and station name is stored in a lookup table.

Critical fields like **Country** and **State** are not present in the daily temperature tables due to normalization, but we can retrieve this information by joining tables.

## UNION

A **UNION** operation is used to append data from multiple tables. For instance, if you want to retrieve temperature recordings for the years 1929 and 1930, you can use a **UNION** to merge the two datasets. A **UNION** combines data vertically, ensuring that the result has the same number, type, and order of columns.

### Daily temperature readings

The diagram illustrates the UNION operation. On the left, two tables are shown: **gsod1929** and **gsod1930**. Both tables have columns: **stn**, **wban**, **temp**, and **year**. The data for both tables is identical, showing three rows for each year (1929 and 1930). A large blue arrow labeled **UNION** points from the two tables to the right, where the resulting combined dataset is shown. This resulting dataset is labeled **gsod1929 UNION gsod1930** and contains all six rows from both tables.

| gsod1929 |       |      |      |
|----------|-------|------|------|
| stn      | wban  | temp | year |
| 030050   | 99999 | 49   | 1929 |
| 030050   | 99999 | 45.7 | 1929 |
| 030050   | 99999 | 48.2 | 1929 |

| gsod1930 |       |      |      |
|----------|-------|------|------|
| stn      | wban  | temp | year |
| 030050   | 99999 | 49   | 1929 |
| 030050   | 99999 | 45.7 | 1929 |
| 030050   | 99999 | 48.2 | 1929 |

| gsod1929 UNION gsod1930 |       |      |      |
|-------------------------|-------|------|------|
| stn                     | wban  | temp | year |
| 030050                  | 99999 | 49   | 1929 |
| 030050                  | 99999 | 45.7 | 1929 |
| 030050                  | 99999 | 48.2 | 1929 |
| 037770                  | 99999 | 50.7 | 1930 |
| 030910                  | 99999 | 56   | 1930 |
| 038560                  | 99999 | 53.2 | 1930 |

- Use **UNION DISTINCT** to remove duplicates or **UNION ALL** if duplicates don't matter.

The diagram shows a SQL query being run against BigQuery. The query uses the `UNION DISTINCT` operator to combine data from two tables: `gsod1929` and `gsod1930`. The query is as follows:

```

gsod1929 UNION gsod1930
stn      wban      temp      year
030050  99999    49       1929
030050  99999    45.7     1929
030050  99999    48.2     1929
037770  99999    50.7     1930
030910  99999    56       1930
038560  99999    53.2     1930
  
```

**UNION DISTINCT removes duplicates whereas UNION ALL keeps every record**

The resulting dataset contains the same data as the UNION example above, but it only includes each unique row once due to the `UNION DISTINCT` clause.

- The schema (number of columns and data types) of the tables must match.
- For large datasets, such as temperature readings from 1929 to 2023, a **wildcard table** can be used to union all tables matching a wildcard expression in BigQuery. This allows you to skip the manual work of typing individual **UNION** operations.

```
#standardSQL
SELECT
  stn,
  wban,
  temp,
  year
FROM
  `bigquery-public-data.noaa_gsod.gsod1929`
  UNION ALL
  `bigquery-public-data.noaa_gsod.gsod1930`
  UNION ALL
  `bigquery-public-data.noaa_gsod.gsod1931`
  UNION ALL
  `bigquery-public-data.noaa_gsod.gsod1932`
# This is getting out of hand...
```



```
#standardSQL
SELECT
  stn,
  wban,
  temp,
  year
FROM
  `bigquery-public-data.noaa_gsod.gsod*`  

# All gsod tables
```

- But what if you want to union from 1929 all the way to 2023, but skip some of the years in between? You can also filter specific years using `_TABLE_SUFFIX`.

Use `_TABLE_SUFFIX` to filter out some of the included tables.

Be as granular as you can.

- e.g. `.gsod2*` instead of `.gsod*` if you only care about the year 2000 onward

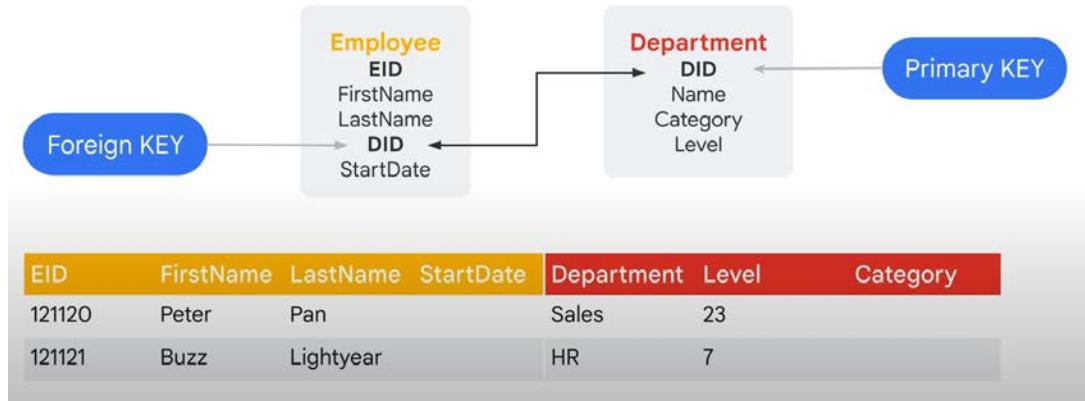
```
SELECT
  stn,
  wban,
  temp,
  year
FROM
  `bigquery-public-data.noaa_gsod.gsod*`  

# All gsod tables after 1950
WHERE _TABLE_SUFFIX > '1950'
```

## JOIN

Let's consider an **Employee** table containing employee information and a **Department** table with department details.

JOINS give you fields from different tables



- The **EID** column in the Employee table uniquely identifies each employee, meaning there can be no duplicates or null values. This is known as the **primary key**—ensuring that no two employees share the same EID.
- Similarly, the **DID** column in the Department table serves as its primary key, as each department is uniquely identified by a non-null ID.
- In the Employee table, **DID** acts as a **foreign key**, meaning its values must match one of the department IDs in the Department table. However, the same DID value may appear multiple times in the Employee table, since multiple employees can belong to the same department.

```
#standardSQL
# Is usaf unique over time?
SELECT
    COUNT(usaf) AS total_count,
    COUNT(DISTINCT usaf) AS distinct_count
FROM
    `bigquery-public-data.noaa_gsod.stations`;
```

| Row | total_count | distinct_count |
|-----|-------------|----------------|
| 1   | 30016       | X 26453        |

No

| Row | usaf   | wban  | name                 | country | state | call | lat    | lon    | elev    | begin    | end      |
|-----|--------|-------|----------------------|---------|-------|------|--------|--------|---------|----------|----------|
| 1   | 912450 | 41606 | WAKE ISLAND AIRFLD   | WQ      | PC    | PWAK | 19.283 | 166.65 | +003.7  | 19451231 | 20100731 |
| 2   | 912460 | 41606 | WAKE ISLAND AIRFIELD | WQ      | UM    | PWAK | 19.283 | 166.65 | +0007.0 | 20100801 | 20170805 |
| 3   | 999999 | 41606 | WAKE ISLAND          | WQ      | PC    | PWAK | 19.283 | 166.65 | +003.7  | 19491031 | 19721231 |
| 4   | 912450 | 99999 | WAKE ISLAND AIRFLD   | WQ      |       |      | 19.283 | 166.65 | +0004.0 | 20000101 | 20100818 |
| 5   | 997387 | 99999 | WAKE ISLAND          | WQ      |       |      | 19.28  | 166.62 | +0005.0 | 20050517 | 20170804 |

- Before we can link the two tables together, we need to find our unique row identifier.
- As you can see from the above query, USAF is not unique.
- The WBAN column also doesn't look unique by itself.

- If neither field is unique on its own, you can concatenate them to create a unique identifier.
- **JOINS** can also be used with multiple conditions, for example, joining on both **USAF** and **WBAN**.

```
#standardSQL
# Is usaf wbnn combo unique over time?
SELECT
    COUNT(CONCAT(usaf,wban)) AS total_stations,
    COUNT(DISTINCT CONCAT(usaf,wban)) AS distinct_stations
FROM
`bigquery-public-data.noaa_gsod.stations`;
```

The screenshot shows a BigQuery results page. At the top, there's a summary table:

| Row | total_stations | distinct_stations |
|-----|----------------|-------------------|
| 1   | 30016          | 30016             |

Below this is a large green "Yes". To the right is a detailed results table:

| Row | usaf   | wbnn  | name                 | country | state | call | lat    | lon    | elev    | begin    | end      |
|-----|--------|-------|----------------------|---------|-------|------|--------|--------|---------|----------|----------|
| 1   | 912450 | 41606 | WAKE ISLAND AIRFLD   | WQ      | PC    | PWAK | 19.283 | 166.65 | +0003.7 | 19451231 | 20100731 |
| 2   | 912460 | 41606 | WAKE ISLAND AIRFIELD | WQ      | UM    | PWAK | 19.283 | 166.65 | +0007.0 | 20100801 | 20170605 |
| 3   | 999999 | 41606 | WAKE ISLAND          | WQ      | PC    | PWAK | 19.283 | 166.65 | +0003.7 | 19491031 | 19721231 |
| 4   | 912450 | 99999 | WAKE ISLAND AIRFLD   | WQ      |       |      | 19.283 | 166.65 | +0004.0 | 20000101 | 20100818 |
| 5   | 997387 | 99999 | WAKE ISLAND          | WQ      |       |      | 19.28  | 166.62 | +0005.0 | 20050517 | 20170604 |

At the bottom left of the results table is a "Table" button, and at the bottom right is a "JSON" button.

If we CONCATENATE the two fields we get a combined unique key showing 30,016 stations.

- When you join two tables, you basically combine data from separate tables that share a common field into one table. In this example, for each row in the temperature table, you try to join with a row from the weather station table if the station number matches the USAF number, AND if the WBAN numbers match.

```

#standardSQL
SELECT
    a.stn,
    a.wban,
    a.temp,
    a.year,
    b.name,
    b.state,
    b.country
FROM
    `bigquery-public-data.noaa_gsod.gsod*` AS a
JOIN
    `bigquery-public-data.noaa_gsod.stations` AS b
ON
    a.stn=b.usaf
    AND a.wban=b.wban

WHERE
    # Filter data
    state IS NOT NULL
    AND country='US'
    AND _TABLE_SUFFIX > '2015'

```

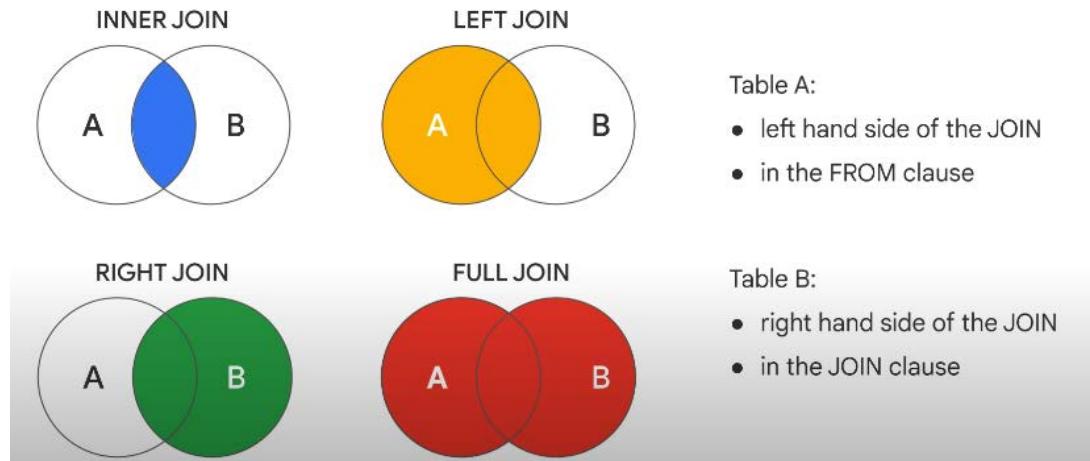
That's called the JOIN condition.

- Aliases are optional in the SELECT statement if the field names are unique between the tables.

|                                   |   |
|-----------------------------------|---|
| Fields from Temperature tables    | #standardSQL<br>SELECT<br>a.stn,<br>a.wban,<br>a.temp,<br>a.year,<br>b.name,<br>b.state,<br>b.country               |
| Fields from Station Details table |   |
| Join type                         | FROM<br>`bigquery-public-data.noaa_gsod.gsod*` AS a<br>JOIN<br>`bigquery-public-data.noaa_gsod.stations` AS b<br>ON |
| Join condition                    | a.stn=b.usaf<br>AND a.wban=b.wban   |
|                                   | WHERE<br># Filter data<br>state IS NOT NULL<br>AND country='US'<br>AND _TABLE_SUFFIX > '2015'                       |

- JOINS can have multiple linking fields to establish the join on logical keys, like the one shown here.

## Types of Joins



- **INNER JOIN:** Returns rows only when there is a match in both tables.
- **LEFT OUTER JOIN:** Returns all rows from the left table and matched rows from the right table.
- **RIGHT OUTER JOIN:** Returns all rows from the right table and matched rows from the left.
- **FULL OUTER JOIN:** Returns all rows from both tables, even if there is no match.
- **CROSS JOIN:** Creates a Cartesian product, returning all possible combinations of rows from both tables.

When using **JOIN**, it's crucial to correctly identify the **primary key** and **foreign key** in your tables. This prevents issues like row duplication or excessive output. For example, in our weather data, we observed how removing a join condition resulted in 128 times more rows, mimicking a **CROSS JOIN**.

```

SELECT
  a.stn,
  a.wban,
  a.temp,
  a.year,
  b.name,
  b.state,
  b.country
FROM
  `bigquery-public-data.noaa_gsod.gsod*` AS a
JOIN
  `bigquery-public-data.noaa_gsod.stations` AS b
ON
  a.stn=b.usaf
  AND a.wban=b.wban

WHERE
  # Filter data
  state IS NOT NULL
  AND country='US'
  AND _TABLE_SUFFIX > '2015'

  15 seconds
  7,176,241 rows returned

```

```

SELECT
  a.stn,
  a.wban,
  a.temp,
  a.year,
  b.name,
  b.state,
  b.country
FROM
  `bigquery-public-data.noaa_gsod.gsod*` AS a
JOIN
  `bigquery-public-data.noaa_gsod.stations` AS b
ON
  a.stn=b.usaf

WHERE
  # Filter data
  state IS NOT NULL
  AND country='US'
  AND _TABLE_SUFFIX > '2015'

  1 minute, 17 seconds
  919,093,886 rows returned

```

That's exactly what you see here. And while the effect is like a cross join, this is a standard join with a poorly chosen key. With this dataset, it results in the same output as if the join were a CROSS JOIN.

### Pitfall: Joining on non-unique keys explodes your data

| gsod2019 |        | stations |        |  |
|----------|--------|----------|--------|--|
| wban     | stn    | wban     | usaf   |  |
| 26656    | 999999 | 24040    | 999999 |  |
| 92828    | 999999 | 53008    | 999999 |  |
| 23906    | 999999 |          |        |  |
| 54918    | 999999 |          |        |  |
| 93804    | 999999 |          |        |  |

| wban  | usaf   |
|-------|--------|
| 26656 | 999999 |
| 26656 | 999999 |
| 92828 | 999999 |
| 92828 | 999999 |
| 23906 | 999999 |
| 23906 | 999999 |
| 54918 | 999999 |
| 54918 | 999999 |
| 93804 | 999999 |
| 93804 | 999999 |

**Understand your data model and relationships**

- Identify your primary and foreign key fields correctly.
- Consider using multiple join conditions if no unique fields exist.
  - Use CONCAT( ) to create composite key fields if no unique fields exist to join on the fields of the composite key, but this may have performance implications.



## Summary



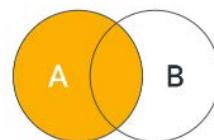
The record identifier needs to uniquely identify a single row.



Spend time exploring the data relationship model between tables.



Use UNION wildcards and \_TABLE\_SUFFIX\_ to quickly add records to a consolidated table.



Use JOINs to enrich data across multiple tables.

## ▼ 3- Cleaning and Transforming your Data

### ▼ Cleaning and Transforming Your Data

Welcome to **Cleaning and Transforming Your Data**. One of the most crucial aspects of data analysis takes place long before you create your first visualization. Although it may not seem as glamorous, data preparation and transformation are essential and cannot be overlooked.



#### SQL + BigQuery UI

- Flexible, fast, and familiar
- Requires SQL knowledge



#### Data preparation tools

- GUI for exploring columns and rows
- Fast summary statistics



#### Visualization tools

- Visually shape and reshape quickly
- See data a different way

In most cases, you will use **SQL** and **data preparation tools** to accomplish this important task, which we will explore in detail throughout this module. Additionally, visualization tools can be valuable in identifying areas where the data requires cleaning or transformation to make it usable. We will discuss this further in a later module.

This module will focus on what constitutes a good dataset and will cover two primary methods for processing, cleaning, and preparing data. The first method involves using **SQL queries in the BigQuery UI**. Additionally, we will briefly introduce other tools available for data preparation, cleaning, and transformation, highlighting when to use each.

## ▼ Five Principles of Dataset Integrity

High-quality datasets lead to high-quality insights. In this lesson, we will discuss the **five principles of dataset integrity** that ensure data quality.

Most data quality discussions revolve around one key idea: "Garbage in, garbage out." If you aim to build accurate models or perform effective data analysis, you must start with clean, high-quality data.

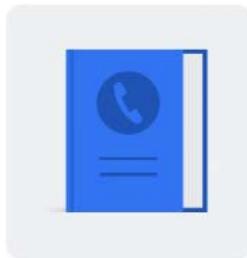
# Garbage in... garbage out



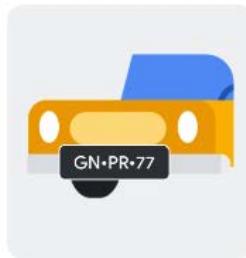
Here are the five key principles that high-quality datasets must follow:

| <b>01</b><br><b>Validity</b>         | <b>02</b><br><b>Accuracy</b>             | <b>03</b><br><b>Completeness</b>           | <b>04</b><br><b>Consistency</b>                    | <b>05</b><br><b>Uniformity</b>             |
|--------------------------------------|--|--|--|--|
| Data conforms to your business rules | Data conforms to an objective true value | Data should be maintained in its full form | Ensures data harmony across many different systems | Data conforms to the same units of measure |

1. **Validity:** The data must conform to your business rules and constraints, such as data type, range, and foreign key relationships. For example, an employee's ID should always follow a specific format and be unique. Ensuring validity prevents inaccurate entries, which often occur in legacy systems or inadequate data-capture methods like spreadsheets.



Phone number



License plate number



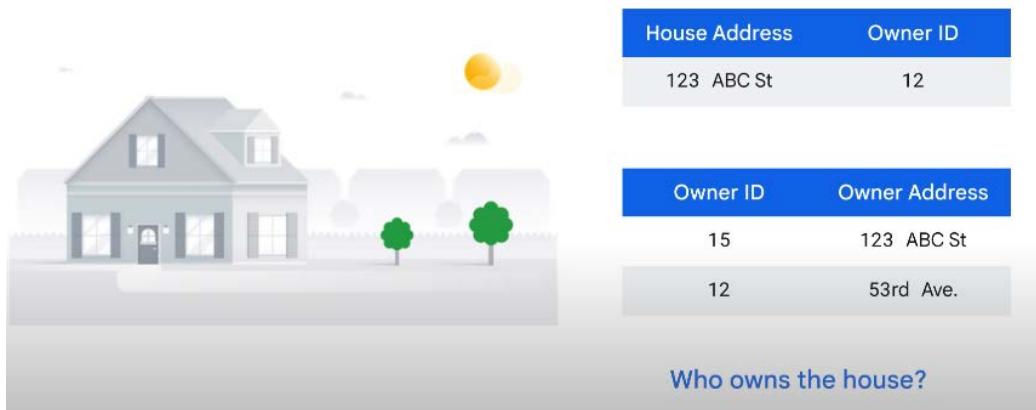
Address

What do these identifiers have in common?  
Why were they set up that way?

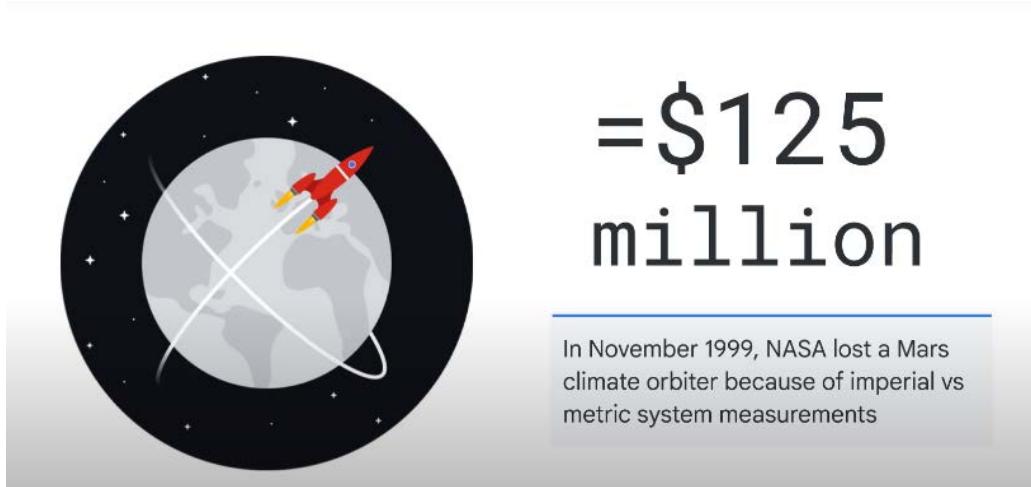
2. **Accuracy:** Data should reflect the true values from a reliable source. For instance, if you have a list of U.S. states, it shouldn't include non-state entries like "Hot Dog." While a string data type may be valid, it won't match the actual set of states. Achieving accuracy often requires cross-referencing data with a trustworthy source of truth, which can be a challenge when "gold standard" data is unavailable.



3. **Completeness:** Your data must represent the entire picture without missing or truncated elements. Incomplete data can lead to biased insights, inaccurate patterns, or overfitting in machine learning models. For example, analyzing only a partial set of sales data may skew the results.
4. **Consistency:** Data must be harmonized across different systems. For instance, a house should not have two different owners listed in separate datasets. Inconsistent data, like this, can lead to referential integrity issues, where two conflicting facts must be reconciled to determine the truth.



5. **Uniformity:** All data must follow the same units of measurement or standards across systems. A famous case of non-uniformity is NASA's loss of a \$125 million Mars orbiter, caused by one team using imperial measurements and another using the metric system. Ensuring uniformity helps avoid such costly errors.



= \$125  
million

In November 1999, NASA lost a Mars climate orbiter because of imperial vs metric system measurements

Maintaining dataset integrity based on these five principles is crucial for generating reliable insights and avoiding common data quality pitfalls.

## ▼ Clean and Transform Data Using SQL

In this lesson, we revisit the **five data integrity rules** discussed earlier and demonstrate how SQL can be used to clean, prepare, and transform your dataset to achieve high data quality. Let's dive into how SQL helps ensure each of these integrity rules.

### 1. Validity

- Set up field data type constraints
- Specify fields as `NULLABLE` or `REQUIRED`
- Proactively check for `NULL` or blank string values
- Check and filter for allowable range values
  - SQL conditionals: `CASE WHEN`, `IF ()`
- Require primary keys / relational constraints in upstream source systems

- **Validity** ensures that your data conforms to business rules and constraints.
- In SQL, you can enforce these rules, such as specifying whether a column can be `NULL` or must contain a value (e.g., `Name` or `ID` fields that should be unique and non-null).

- This can be done while creating tables to ensure incoming data is valid, or you can validate data after it's inserted using **conditional logic** like `IF THEN ELSE` and `CASE WHEN` statements.

## 2. Accuracy

- Create test cases or calculated fields to check values
  - SQL: `(quantity_ordered * item_price) AS sub_total`
- Look up values against an objective reference dataset
  - SQL: `IN( )` with a subquery or JOIN
- **Accuracy** ensures the data makes logical sense.
- You can create **calculated fields** to test whether values are accurate, or compare the data to a known reference dataset, such as checking states in the U.S.
- Use SQL constructs like **JOINS**, **subqueries**, and **CTEs** to validate the occurrence of values with the `IN` operator, ensuring data is correct.

## 3. Completeness

- Thoroughly explore the existing dataset shape and skew and look for missing values, NULLs, or blank strings
  - SQL: `NULLIF()`, `IFNULL()`, `COALESCE()`
- Enrich the existing dataset with others using UNIONs and JOINs
  - SQL: UNION, JOIN
  - Example: Multiple years of historical data are available for analysis
- **Completeness** checks if any data is missing.
- Missing data can be handled using functions like `COALESCE`, which substitutes a value if `NULL` is found.
- You can also use **JOINS** and **UNIONs** to merge datasets and ensure completeness by combining and enriching data sources.

## 4. Consistency

- Store one fact in one place and use IDs to look up
- Use string functions to clean data
  - PARSE\_DATE()
  - SUBSTR()
  - REPLACE()

- **Consistency** requires the uniformity of your data. If you're working with inconsistent data like dates, SQL provides string functions like `SUBSTRING` and `REPLACE` to standardize formats across datasets.
- You don't need to modify the raw data. These steps can be part of a repeatable data pipeline to handle new data as it arrives.

## 5. Uniformity

- Document and comment your approach.
- Use `FORMAT()` to clearly indicate units.
- `CAST()` data types to the same format and digits.
- Label all visualizations appropriately.
- `DATE()`, `TIME()` and `DATETIME()` all accept a timezone parameter.
- Convert unix epoch values using `TIMESTAMP_MICROS()`, `TIMESTAMP_MILLIS()`, `TIMESTAMP_SECONDS()`

- **Uniformity** ensures that data types are standardized.
- Functions like `FORMAT` and `CAST` can be used to ensure all data is consistent, such as converting between metric and imperial units.
- For date and time uniformity, functions like `DATE()`, `TIME()`, and `DATETIME()` allow easy conversion between time zones, and converting Unix epoch times to a standard timestamp ensures consistency.

### Example Query

Here's an example where we pull weather data and filter out rows where the U.S. state is either `NULL` or a blank string:

```

SELECT * FROM
`data-to-insights.demos.gsod_stations`
WHERE state IS NOT NULL
LIMIT 10

```

Why does this query still show blank state values when we clearly filtered on IS NOT NULL?

| Row | usaf   | wban  | name       | country | state | call | lat  | lon | elev | begin    | end      |
|-----|--------|-------|------------|---------|-------|------|------|-----|------|----------|----------|
| 1   | 007011 | 99999 | CWOS 07011 |         |       | null | null |     |      | 20120101 | 20121129 |
| 2   | 007005 | 99999 | CWOS 07005 |         |       | null | null |     |      | 20120127 | 20120127 |
| 3   | 007023 | 99999 | CWOS 07025 |         |       | null | null |     |      | 20120127 | 20120127 |
| 4   | 007044 | 99999 | CWOS 07044 |         |       | null | null |     |      | 20120127 | 20120127 |
| 5   | 007047 | 99999 | CWOS 07047 |         |       | null | null |     |      | 20120613 | 20120717 |
| 6   | 007083 | 99999 | CWOS 07083 |         |       | null | null |     |      | 20120713 | 20120717 |
| 7   | 007033 | 99999 | CWOS 07034 |         |       | null | null |     |      | 20121024 | 20121106 |
| 8   | 007084 | 99999 | CWOS 07084 |         |       | null | null |     |      | 20121214 | 20121217 |
| 9   | 007094 | 99999 | CWOS 07094 |         |       | null | null |     |      | 20121217 | 20121217 |

In this case, an empty string (a valid data value) is different from `NULL`, which is the absence of data. This query filters out both nulls and empty strings to ensure clean data.

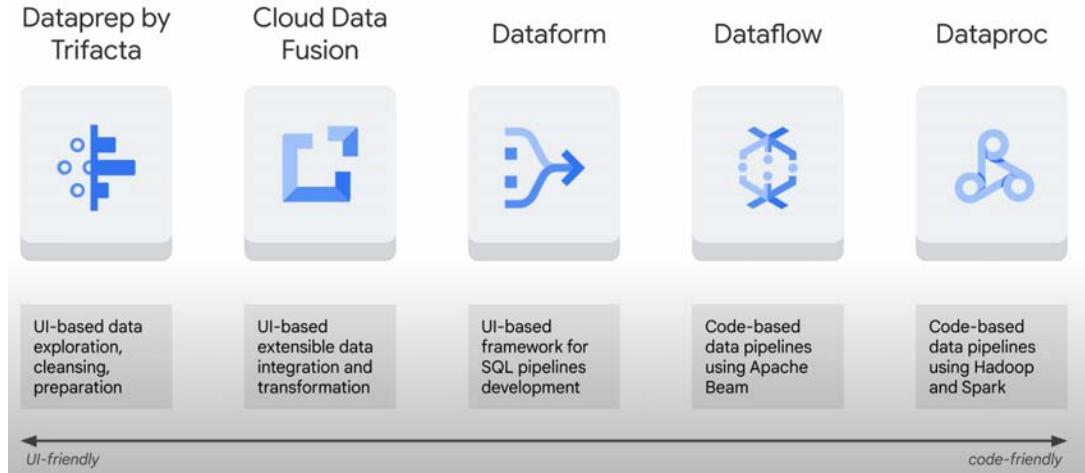
Remember: An empty string or blank string is a valid data value whereas a `NULL` is the absence of any data. A valid null value in BigQuery looks exactly like what these latitude and longitude columns look like.

How could we adjust this query to filter out blanks as well?

- You would have to update your WHERE clause like how it is shown in the slide, where you also filter out blank strings in addition to null values.

## ▼ Clean and Transform Data Using SQL - Other Options

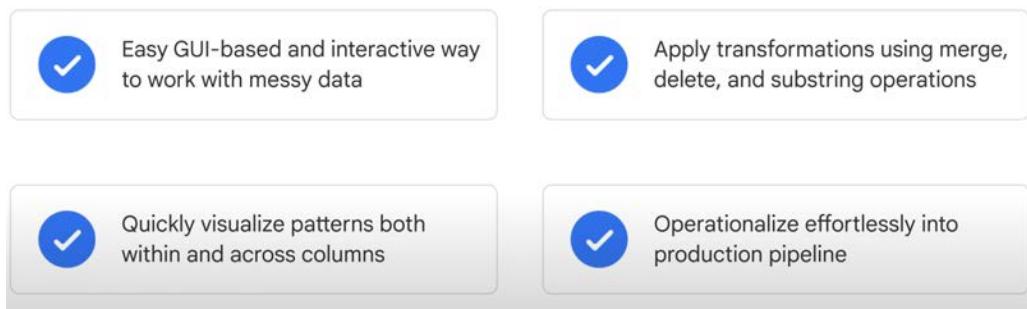
In this lesson, we explore alternative tools and services to clean, prepare, and transform data beyond SQL. The focus will be on several Google Cloud services that offer unique features for data transformation.



## 1. Cloud Data Fusion

- A fully managed, cloud-native service for building and managing data pipelines.
- It uses a no-code graphical interface to help users like business professionals, data analysts, and scientists create data pipelines.
- **Wrangler tool:** A key feature of Cloud Data Fusion, Wrangler allows users to interactively clean, transform, and enrich a small sample (up to 10 MB) of their data before applying transformations to the entire dataset.

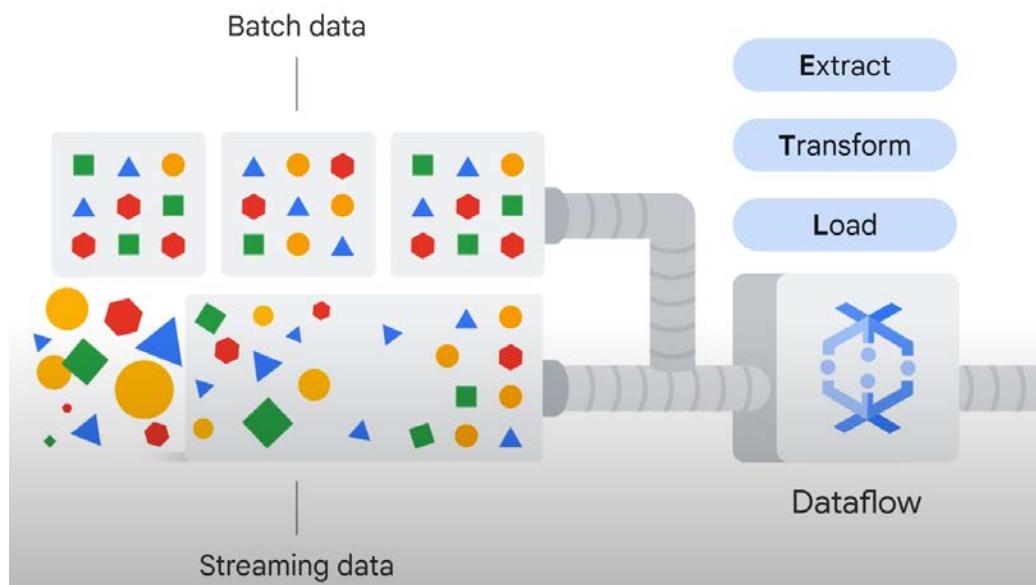
## Wrangler capabilities



- Once satisfied with the transformations, users can operationalize the data pipeline and send transformed data to targets like BigQuery.

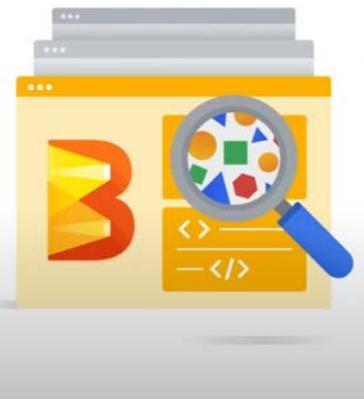
## 2. Dataflow

- Dataflow creates a pipeline to process data:



- A Google Cloud service that supports unified stream and batch data processing at scale, allowing for sophisticated ETL (Extract, Transform, Load) workflows.
- Based on the open-source **Apache Beam** project, Dataflow simplifies the mechanics of large-scale data processing by letting developers focus on the logic of the data pipeline, while the service handles the orchestration of parallel processing and distribution.
  - What is Apache Beam ?

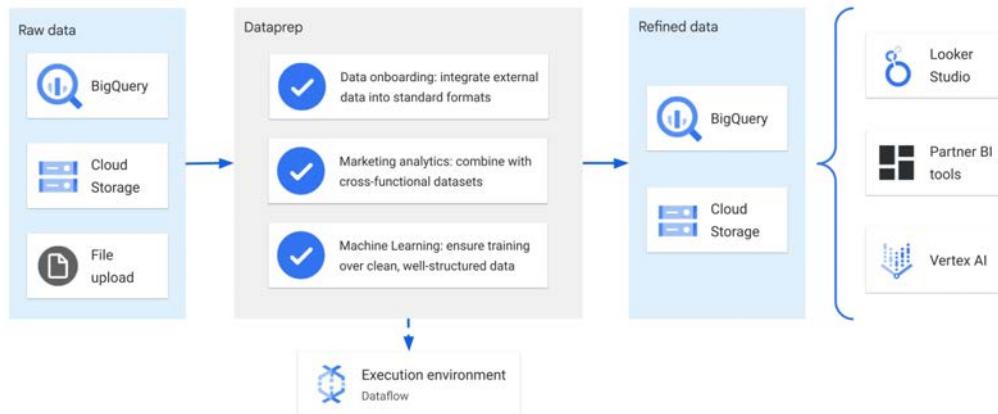
- Apache Beam is an open source unified programming model to define both **batch** and **streaming** data-processing pipelines used by Dataflow.
- Beam SDKs** are used to create the pipeline in the programming language of your choice.
- Pipelines can be run **locally** on your machine or on other **backend services** with their own features.
- A **Runner** is used to execute your pipeline on a backend of your choice.
- Dataflow** is one of the runners available in Beam.



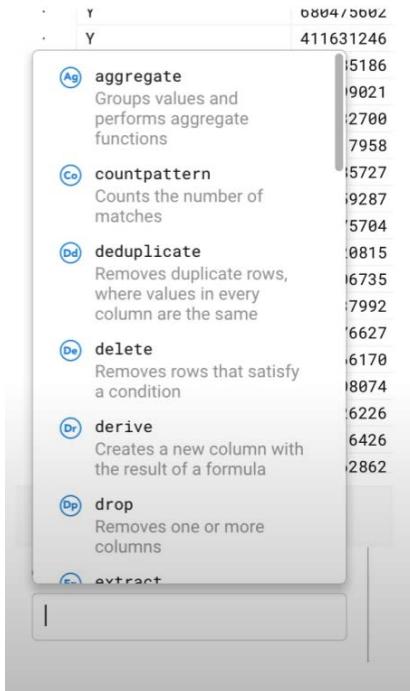
- Ideal for pipelines that require moving, transforming, and enriching data or joining data from multiple sources.

### 3. Dataprep

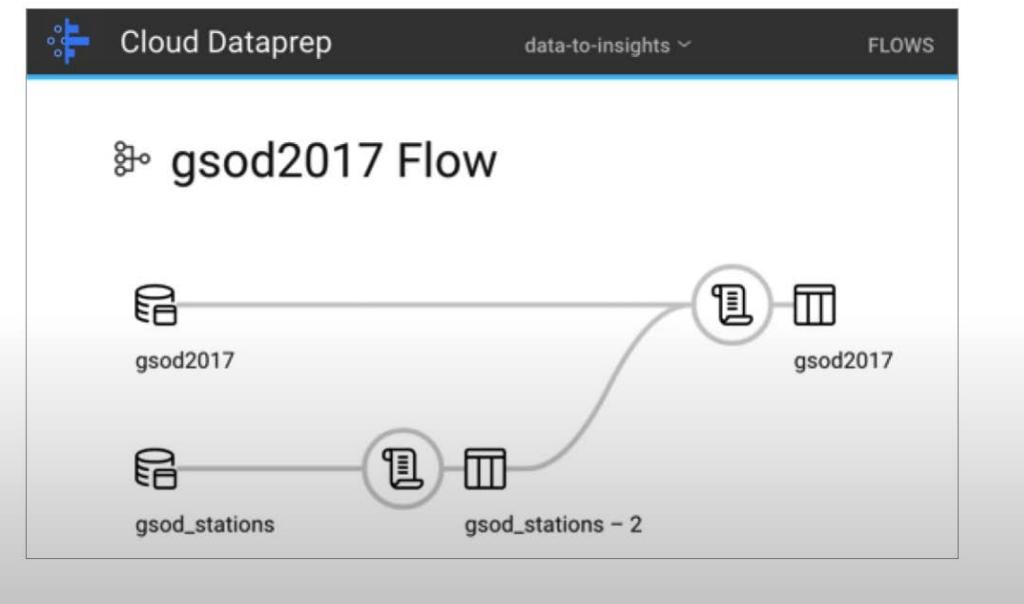
- Cloud Dataprep by Trifacta is a visual data preparation tool running on Dataflow



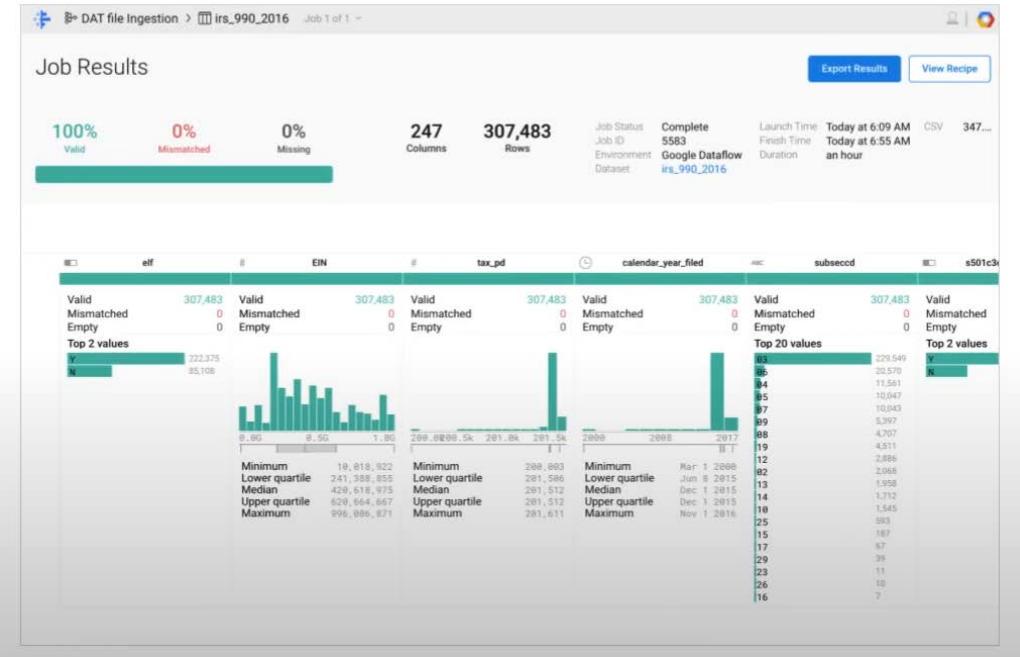
- A graphical user interface (GUI) tool powered by Trifacta and built on Dataflow, allowing users to perform data cleaning and preparation on massive datasets without having to manage infrastructure.
  - Integrated with Google Cloud, Dataprep can access data from BigQuery, Cloud Storage, and more, supporting formats like CSV, JSON, and relational tables.
  - Refined data can be ported directly to tools like BigQuery, Looker Studio, or Vertex AI.
- You can pick from predefined wranglers to process your data.



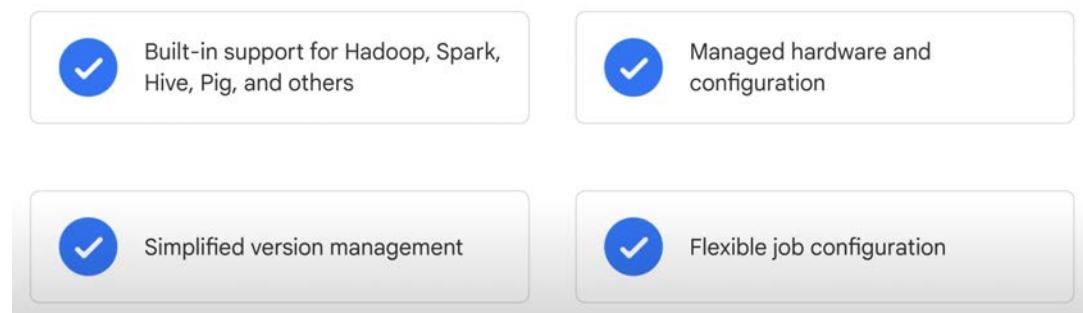
- Review the final data flow and then run the job, which will then kick off a new Dataflow job behind the scenes.



- Use dashboards to monitor your jobs, and dig into the metrics for your transformed datasets.



## 4. Dataproc



- A fully managed service for running Apache Hadoop, Apache Spark, and other big data frameworks.
- Dataproc integrates with Google Cloud services like BigQuery, Cloud Storage, and Cloud Bigtable, making it a comprehensive platform for modernizing data lakes, ETL processes, and secure data science.
- Enterprises can autoscale Dataproc clusters to support a variety of batch, stream, and machine learning workloads.

## 5. Dataform

- Available through the BigQuery console, Dataform helps develop and operationalize scalable data transformation pipelines using SQL.
- We'll cover this in greater detail in a later module.

### Conclusion:

## When to use what

| Product              | Use case   | Integrations   | Open Source                  | Persona                      | Notes   |
|----------------------|--|--|------------------------------|------------------------------|---|
| Dataprep by Trifecta | Data wrangling   | Google Cloud Storage, BigQuery, other connection types | -                            | Data Analyst, Data Engineer  | <ul style="list-style-type: none"> <li>Not a Google-built product</li> <li>Leverages Dataflow for runtime</li> </ul>    |
| Cloud Data Fusion    | Enterprise data integration for data warehouses and data marts | hybrid, multi-cloud                                    | CDAP                         | Data Engineer, ETL developer | <ul style="list-style-type: none"> <li>Supports only ETL (as of now)</li> <li>Leverages Dataproc for runtime</li> </ul> |
| Dataform             | SQL transformation pipelines development and management        | BigQuery   | Dataform Core                | Data Analyst, Data Engineer  | <ul style="list-style-type: none"> <li>serverless framework</li> <li>integrates with GitHub and Gitlab</li> </ul>       |
| Dataflow             | Streaming analytics and ETL workloads                          | Pub/Sub, Google Cloud Storage, BigQuery, other OSS     | Apache Beam                  | Data Engineer, ETL developer | <ul style="list-style-type: none"> <li>serverless streaming platform</li> <li>ETL/Batch is supported</li> </ul>         |
| Dataproc             | ETL workloads  | Google Cloud Storage, BigQuery, other OSS              | Apache Hadoop, Spark, others | Data Engineer, ETL developer | <ul style="list-style-type: none"> <li>managed service</li> <li>Dataproc Serverless for Spark available</li> </ul>      |

These services provide flexibility in terms of code (Dataflow, Dataproc) or no-code (Cloud Data Fusion, Dataprep) solutions to clean and transform data. Each service has its unique strengths, making them suitable for different scenarios, from simple transformations to complex ETL processes.

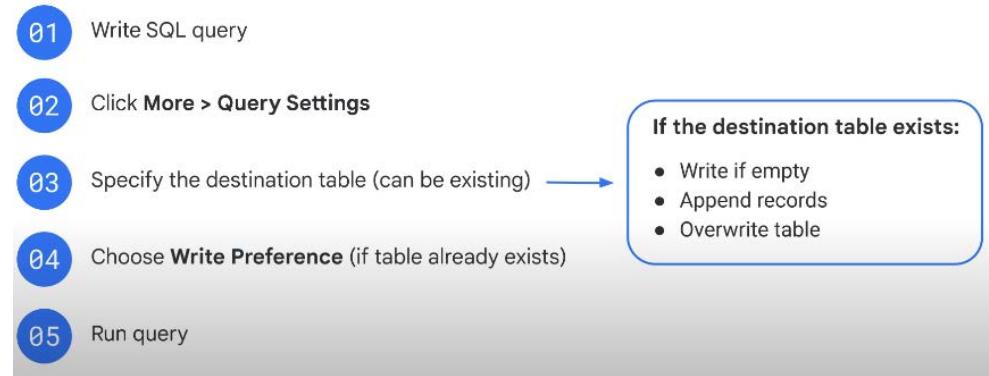
## ▼ 4- Ingesting and Storing New BigQuery Datasets

### ▼ Permanent versus Temporary Tables

When running queries in BigQuery, query results are saved either in **permanent** or **temporary** tables. Here's an overview of how they function:

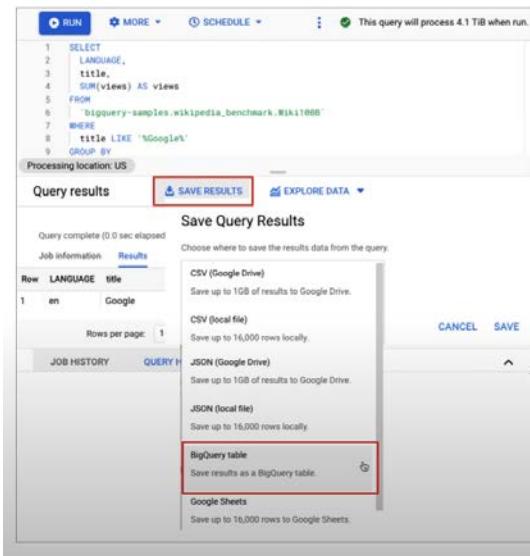
#### 1. Permanent Tables

- **Permanent tables:** You can specify a destination table before running a query. This can be an existing table where you choose to append or overwrite data, or a new table if none exists. Data saved to permanent tables is retained for long-term use and can be queried multiple times.
- **Usage:** Data analysts often run SQL queries on raw data sources and save results in **new reporting tables** for future analysis.
  - How to create a new permanent table



## 2. Temporary Tables

- **Temporary tables** are automatically created by BigQuery when you run a query without specifying a destination table.
- **Characteristics:**
  - Data in temporary tables is available for **24 hours**.
  - After running a query, you can save the results in a permanent table by clicking **Save Results** and selecting **BigQuery table**.



- Query cache results are stored as temporary tables. Cached results are reused in subsequent queries if the underlying data hasn't changed and no non-deterministic functions (like `CURRENT_DATE`) are used.

## 3. Views

- A **view** is a virtual table defined by a SQL query but does not store the results.

- Storing results in view

The screenshot shows the Google Cloud BigQuery interface. In the top right corner of the query editor, there is a context menu with options: 'Save Query', 'Save View' (which is highlighted with a red box), and 'Save as...'. Below the editor, the results pane displays a table with three rows of data. The table has columns: Row, LANGUAGE, title, and views. The data is as follows:

| Row | LANGUAGE | title        | views    |
|-----|----------|--------------|----------|
| 1   | en       | Google       | 97917790 |
| 2   | es       | Google       | 21972680 |
| 3   | en       | Google_Earth | 15973740 |

- A view is a saved SQL query (a virtual table)
- The underlying query is executed each time the view is accessed
- When you select from a view, you are executing the query that defines the view, and it fetches data from the underlying tables in real time.
- Creating Views:
  - Use the `CREATE VIEW` statement in SQL.

```
CREATE VIEW ecommerce.shirts_vw AS
SELECT DISTINCT
    v2ProductName
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE LOWER(v2ProductName) LIKE '%shirt%'
LIMIT 100
```

- You can use the `OR REPLACE` clause to overwrite the query of an existing view or use `IF NOT EXISTS` to create the view only if a view by that name doesn't already exist.

```
CREATE OR REPLACE VIEW ecommerce.shirts_vw AS
SELECT DISTINCT
    v2ProductName
FROM
    `data-to-insights.ecommerce.all_sessions`
WHERE LOWER(v2ProductName) LIKE '%shirt%'
LIMIT 100
```

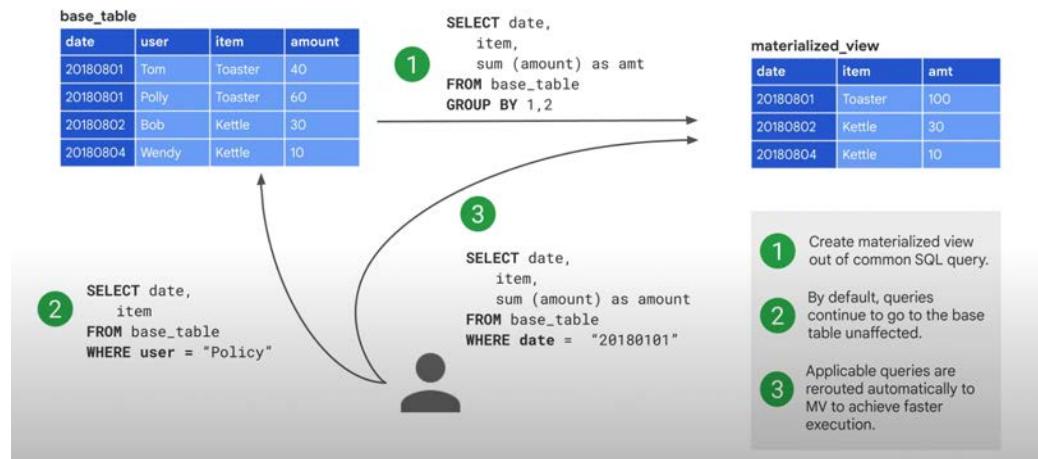
- This discussion would not be complete without showing you the DDL syntax to create a new table schema.

```
CREATE [OR REPLACE] TABLE Dataset.TableName [IF NOT EXISTS] (
    Column1 INT64,
    Column2 STRING,
    Column3 BOOLEAN NOT NULL
    ...
);
```

- This will create an empty table in the specified dataset, using the specified name, with the corresponding columns and data types.
- And in the future you can use the INSERT statement (which is a DML, or data manipulation language) to insert rows into the table.
- You can also use a load job from the command line to populate the table with data coming from a CSV file, for instance.

## 4. Materialized Views

- **Materialized views** cache the results of a query and are periodically refreshed, offering better performance for repetitive and common queries.



- Use the **CREATE MATERIALIZED VIEW** DDL statement to create materialized views.

```
CREATE MATERIALIZED VIEW movielens_demo.recent_movie_ratings
AS
(
  SELECT
    movieId,
    AVG(rating) AS avg_rating,
    COUNT(rating) AS no_rating
  FROM `movielens_demo.ratings`
  WHERE timestamp > 874355425 # 1997/9/15 20:30:25 UTC
  GROUP BY movieId
)
```

- **Benefits:**
  - Queries using materialized views are generally **faster** and consume fewer resources.
  - BigQuery **automatically refreshes** materialized views within 5 minutes of changes to the base table, but not more frequently than every 30 minutes.
  - Materialized views can be manually refreshed or partitioned if the base table is partitioned.
- **Efficiency:** When querying a materialized view, BigQuery reads only the **delta changes** from the base table, ensuring up-to-date results with optimized performance.

Materialized views are ideal for workloads with frequent, repetitive queries, providing a significant boost in query performance and resource efficiency.

## ▼ Ingesting New Datasets

In the previous section, we covered how to save the results of running SQL queries on BigQuery tables into other permanent BigQuery tables for further analysis. But what happens when the source data is not yet available in BigQuery?

In this section, we will explore various methods for loading data into BigQuery. The method you choose will depend on the level of transformation needed.

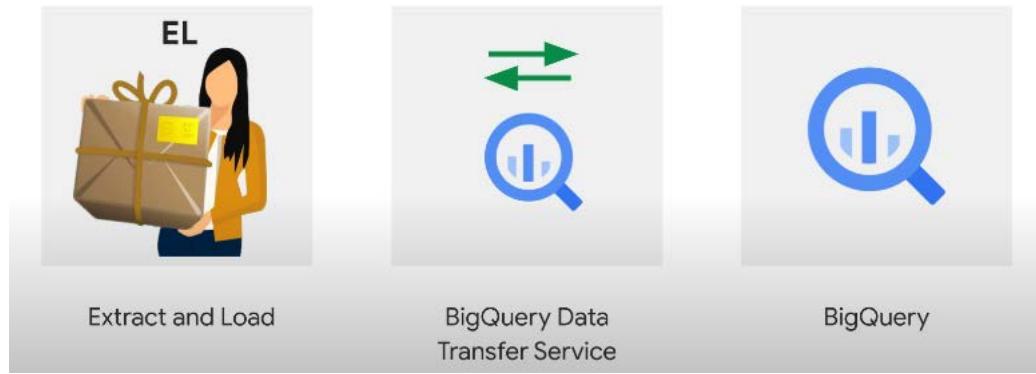
### Data Loading Methods:



- **EL (Extract and Load):** This approach is used when data can be imported as-is, meaning the source and target have the same schema.
- **ELT (Extract, Load, Transform):** In this case, raw data is loaded directly into the target, and the transformations are applied afterward within the target.
- **ETL (Extract, Transform, Load):** Here, data is transformed before it is loaded into the target, typically using an intermediate service.

Let's take a closer look at these methods:

#### 1. Extract and Load (EL):



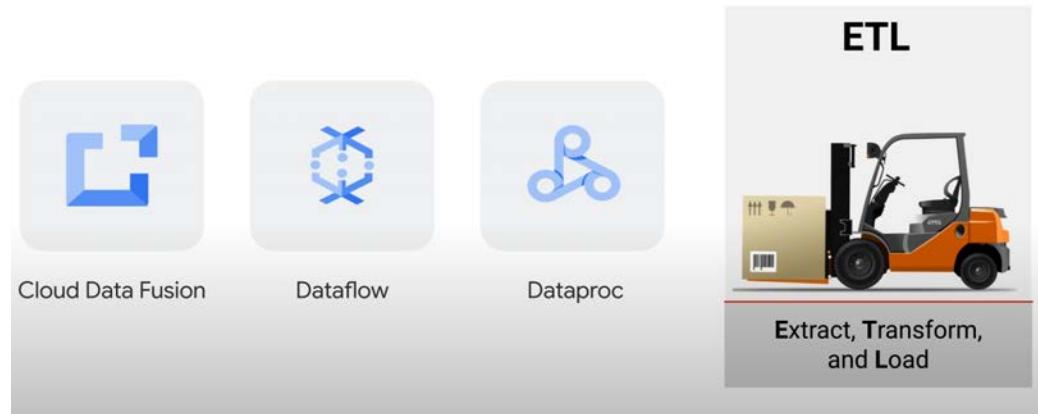
EL is the simplest method. If the data is usable in its original form, there's no need for transformation. You just load it into BigQuery. In the hands-on lab at the end of this module, you'll get the chance to practice loading data from different sources into BigQuery. You can also use BigQuery Data Transfer Service to automate the data ingestion process on a scheduled, managed basis.

### 2. Extract, Load, Transform (ELT):



Sometimes, the raw data in its original form is not immediately usable, and transformations need to be applied. In such cases, you can load the data into BigQuery and use your SQL expertise to build and apply the necessary transformations. The ELT approach is becoming the core of modern data pipelines, with many organizations shifting from an ETL approach, which relies on external tools like Spark, to an ELT approach that utilizes SQL pipelines directly in BigQuery.

### 3. Extract, Transform, Load (ETL):



When the transformations required are complex or numerous, ETL may be the preferred approach. This is especially true if you need to apply transformations before loading data into BigQuery. In such cases, tools like Cloud Data Fusion, Dataflow, or Dataproc can be used to build transformation pipelines before the data is loaded into BigQuery.

## BigQuery Data Ingestion Options



- **Format Flexibility:**

BigQuery can ingest datasets from a wide range of formats. Once the data is inside BigQuery's native storage, it is fully managed by the BigQuery team at Google. This includes features like replication, backups, scaling, and more.

- **External Data Queries:**

You can also bypass BigQuery-managed storage by querying external data sources directly.

- **Streaming Data:**

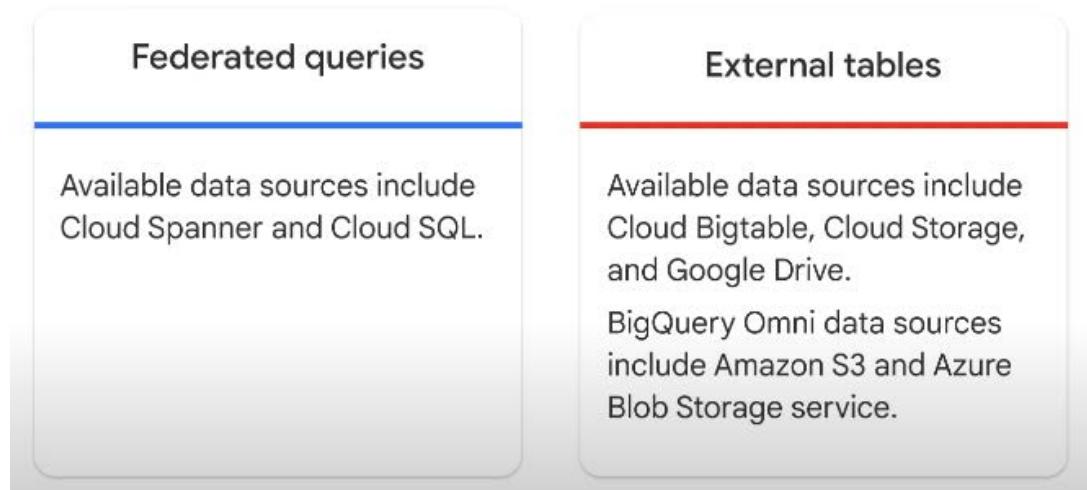


Another option is to stream records into BigQuery using tools like Dataflow or Data Fusion, or through the API. A common use case is event logging, where applications stream user interactions or system events in real time to BigQuery for analysis. This data can be used to monitor trends, detect areas of high interaction, or catch error conditions.

## ▼ External Data Sources

- **External data sources** allow querying data directly from BigQuery even if the data is stored outside BigQuery.
- Examples include data from:
  - Other Google Cloud databases.
  - Files in **Cloud Storage**.
  - Data in other cloud services (e.g., AWS, Azure) without migration.

## Mechanisms to Query External Data

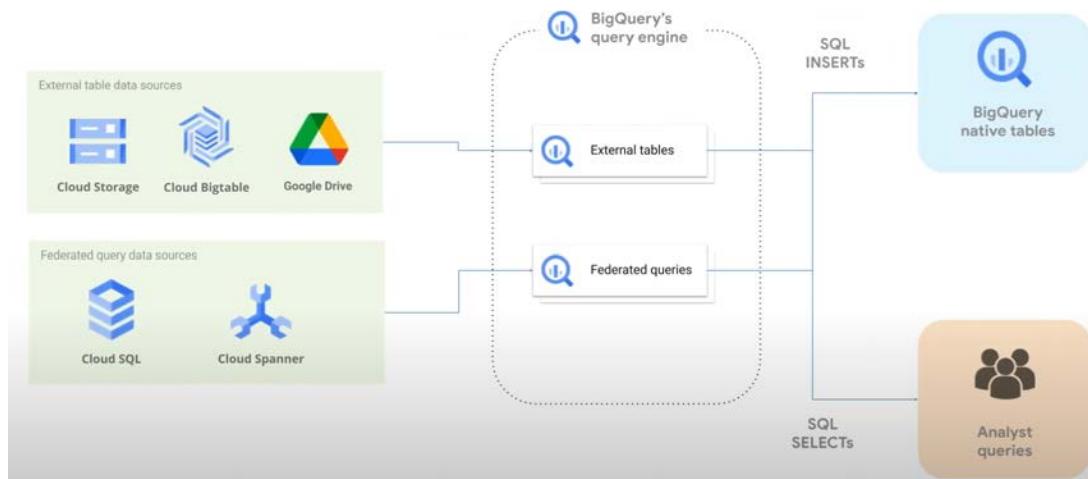


## 1. External Tables

## 2. Federated Queries

We'll also touch on tools like **BigQuery Omni** and **BigLake tables**, which allow access to data in **AWS buckets** and **Azure blob storage**.

## Use Cases for External Data Sources:



- **Ad-hoc queries** where performance isn't critical.
- Accessing **short-term** or **infrequently accessed** data.
- Accessing **constantly changing data** (e.g., from Google Sheets).
- Joining BigQuery tables with **dynamic external data** without reloading the data into BigQuery.

## ELT Workloads

- Extract-Load-Transform (ELT) workflows allow you to load and clean data directly in BigQuery using `CREATE TABLE ... AS SELECT` queries.
- You can also access data in **AWS buckets** or **Azure blob storage** with **BigQuery Omni** remote tables.

## External Tables vs. Federated Queries

|                         | External tables  | Federated queries   |
|-------------------------|--|---|
| Compatible data sources | <ul style="list-style-type: none"> <li>✓ Cloud Storage</li> <li>✓ Cloud Bigtable</li> <li>✓ Google Drive</li> </ul>                  | <ul style="list-style-type: none"> <li>✓ Cloud SQL</li> <li>✓ Cloud Spanner</li> </ul>  |
| Metadata                | <ul style="list-style-type: none"> <li>✓ Metadata and schema stored in BigQuery storage.</li> </ul>                                  | <ul style="list-style-type: none"> <li>✗ No metadata in BigQuery storage</li> <li>✓ <a href="#">Query metadata workaround</a></li> </ul>                      |
| Access controls         | <ul style="list-style-type: none"> <li>✓ Can set <a href="#">table access controls</a> for row- or column-level security.</li> </ul> | <ul style="list-style-type: none"> <li>✗ No access controls in BigQuery. <a href="#">Queries use a service account</a> in the external connection.</li> </ul> |
| Partitioning            | <p>External Tables are read-only but external partitions <a href="#">can be queried</a>.</p>   | <p>Federated queries must be read-only. Querying partitions must follow target semantics.</p>   |

## BigQuery Omni

- A specialized type of external table that allows querying data across clouds, like **S3** and **Azure Blob Storage**.

## Considerations for External Tables

- **Performance:** Querying external tables is **slower** than querying native BigQuery tables due to data being located outside BigQuery.
  - No **caching** or **query size estimates**.
  - **Query performance** depends on the external storage type (e.g., **Cloud Storage** is faster than **Google Drive**).
- **Data Consistency:** BigQuery does not guarantee data consistency for external sources, meaning changes to the underlying data during a query could cause unexpected results.
- **Limitations:**
  - BigQuery **cannot export data** from external sources.
  - External data cannot be used in **wildcard table queries**.

## Conclusion

- While BigQuery allows direct querying of external data sources for convenience, loading data into BigQuery storage is recommended when performance and speed are important.
- We can stream individual records into BigQuery using other products or APIs for real-time updates.

In the next session, we'll practice **creating new datasets**, **loading external data**, and **running queries** on this new data.

## ▼ 5- Visualizing Your Insights from BigQuery

### ▼ Data Visualization Principles

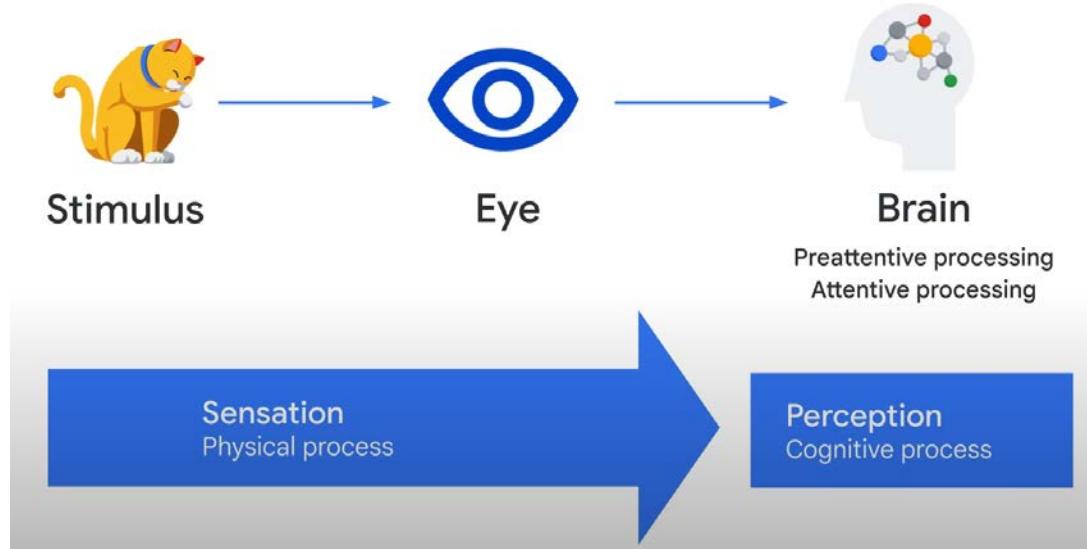


1. **Easier Trend Detection:** Visualization tools reveal hidden trends more effectively than raw data, enabling users to spot patterns at a glance.
2. **Interactive Exploration:** Users can interact with visualizations, such as drilling down into specific time-series anomalies to explore details visually.
3. **Effective Storytelling:** Visualizations help tell cohesive data-driven stories, making insights clearer and more engaging for the audience.
4. **Efficient Summaries:** Dashboards and reports simplify large datasets, turning billions of rows into easy-to-digest summaries, which support quick decision-making.

### BigQuery and Visualization Tools

- Visualization tools like **Looker Studio** can leverage BigQuery's performance benefits, allowing visualizations to be built on top of data without requiring SQL knowledge.

### Visualization Theory: Art and Science



- Data visualization is both an art and a science. It involves how the brain perceives and processes information visually. When you see a visualization, it's not just your eyes processing the data—your brain is also working to interpret it. For instance, your eyes might recognize a common object like a cat instantly because your brain has already formed intuitive connections.
- **Preattentive Processing:**
  - Automatic, unconscious recognition of certain visual elements without focused attention.
  - Helps in quickly identifying patterns or anomalies, reducing cognitive load.

## Examples of Preattentive Processing in Visuals

- **Crowded Visual:** When asked to identify the number "5" among a crowded visual, users may struggle to spot them quickly without any visual aid.

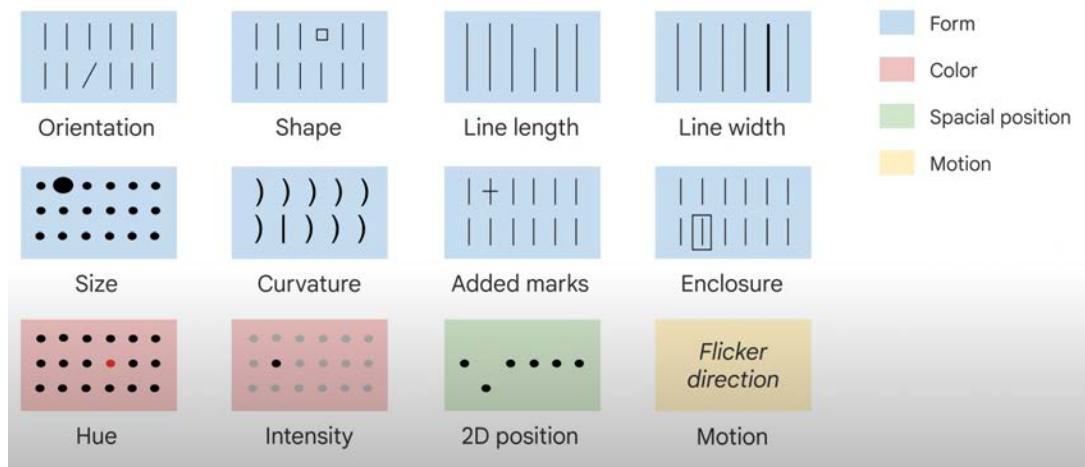
```
69750429347493732418605783578  
58728294974654487818676453214  
24439684634233529867321903875  
65878893745390932975659391732  
14725920189374476564722175652
```

- **Enhanced Visual:** By bolding the number "5", it becomes easier to identify the target due to visual contrast.

```
69750429347493732418605783578  
58728294974654487818676453214  
24439684634233529867321903875  
65878893745390932975659391732  
14725920189374476564722175652
```

**Visual Encoding:** Techniques like bolding, highlighting, and adjusting shapes or colors make it easier to focus on key information, allowing the brain to process the data more quickly.

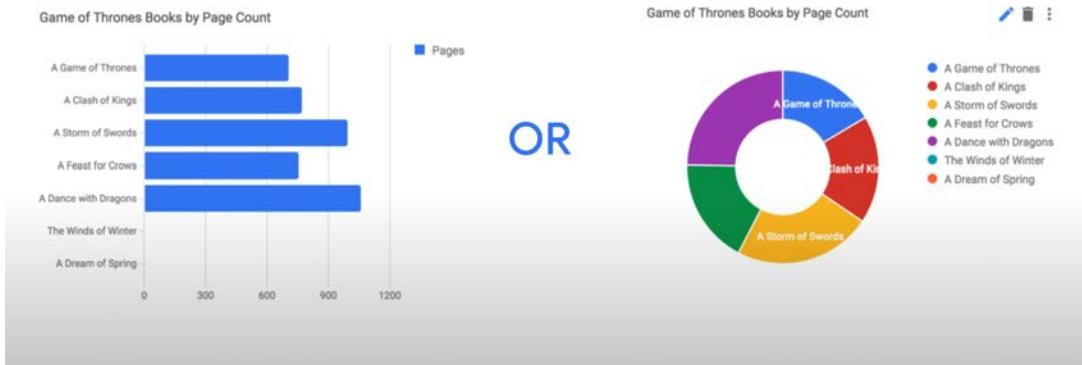
### Key Visualization Elements:



- **Orientation:** Adjusting how elements are aligned to guide attention.
- **Shape/Skew:** Using distinct shapes to distinguish categories.
- **Size:** Larger elements attract more attention.
- **Color:** Contrast, hue, and saturation can guide focus.
- **Motion:** Movement can direct attention to specific changes over time.

## Choosing Effective Visuals





- 1. Donut vs. Line Graph:** Use donut charts for comparing proportions, such as characters by gender, while line graphs are better for time-series data.
- 2. Bar Chart vs. Donut Chart:** Bar charts handle multiple dimensions more effectively, especially when comparing items with valid "0" values. Donut charts can become crowded with too many categories.

## Dimensions vs. Measures

| Description   |                     |  | Examples   |
|---------------|---------------------|--|--|
| 1. Dimensions | Qualitative values  |  | <ul style="list-style-type: none"> <li>Name</li> <li>Location</li> <li>Part number #</li> <li>Job title</li> </ul> |
| 2. Measures   | Quantitative values |  | <ul style="list-style-type: none"> <li>Total revenue</li> <li>Average salary</li> <li>Count of errors</li> </ul>   |

- Dimensions:** Qualitative values like names, dates, and geographic data used to categorize and segment data.
- Measures:** Quantitative values (e.g., count, average) derived from dimensions, used to quantify data for analysis.

## Looker Studio Example



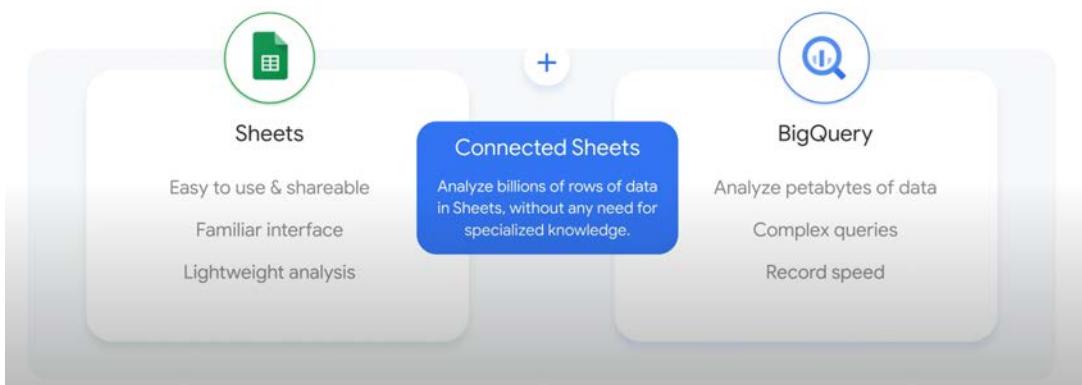
- Looker Studio allows for easy creation of visualizations using dimensions and measures.
- **Visual Encoding** in reports (e.g., histograms, line charts, maps) helps users understand the overall picture through preattentive processing.

## Role of Reports:

- Visualize data.
- Restrict access to specific data.
- Share insights with viewers and editors.
- Present data in a way that supports decision-making.

## ▼ BigQuery Connected Sheets

**Connected Sheets** brings the power of Google Cloud together with the ease and familiarity of Google Workspace, specifically Google Sheets. This tool allows users to analyze large-scale data stored in BigQuery directly through a spreadsheet interface, making big data analysis more accessible.



## Key Benefits of Connected Sheets:

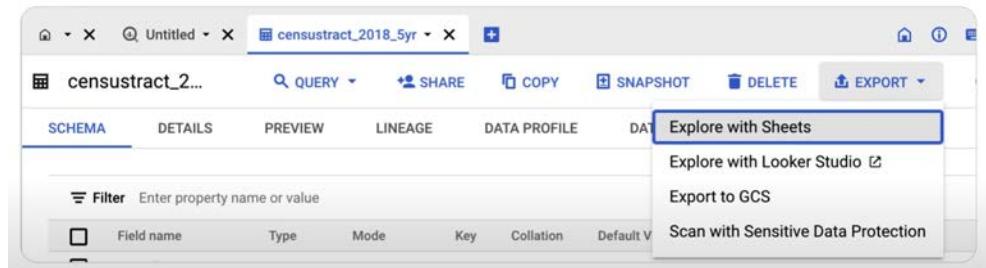
- **Democratizing Big Data:** Users from across the organization, not just data scientists, can access, analyze, and visualize large datasets without needing to write SQL queries or use specialized tools.
- **Reduced Workload for Specialists:** By empowering non-technical users to work with data independently, the reliance on specialized roles, such as data analysts and scientists, is minimized. This leads to a more efficient use of skilled resources.
- **Scalable Analysis:** Unlike traditional spreadsheets, Connected Sheets allows you to analyze billions of rows of data from BigQuery without performance issues.
- **Familiar Tools:** Users can use the same tools they are comfortable with, such as pivot tables, charts, and functions, while accessing powerful cloud data.
- **Secure and Centralized Data:** Data remains secure in BigQuery, and there are no risks associated with exporting data to individual machines. This reduces the risk of data manipulation or loss.

## Key Features:

1. **Access BigQuery Data in Sheets:**

1. Start with a blank Google Sheet
2. Choose **Data > Data Connectors > Connect to BigQuery**
3. Select a project
4. Then choose a dataset
5. Select the table and connect
6. Have fun analyzing the data!

- Start with a blank Google Sheet.
- Go to the top menu, select **Data > Data Connectors > Connect to BigQuery**.
- Choose the relevant project, dataset, and table, and the data will populate in your sheet.
- You can also set up a connected sheet directly from the BigQuery web UI by selecting **Explore with Sheets** from the **Schema** or **Export** options.



- If you want to only see selected rows based on a query, you can again use the BigQuery.
- UI, run the SQL query, and then use the Explore data dropdown button and select Explore with Sheets.

Untitled 2

**RUN** **SAVE** **SHARE** **SCHEDULE** **MORE**

```

1 SELECT
2   *
3   FROM
4   `qwiklabs-gcp-04-b0173b128511.public_sector.censustract_2018_5yr`
5   WHERE
6   total_pop > 10000

```

Press Option+F1 for Accessibility

**Query results**

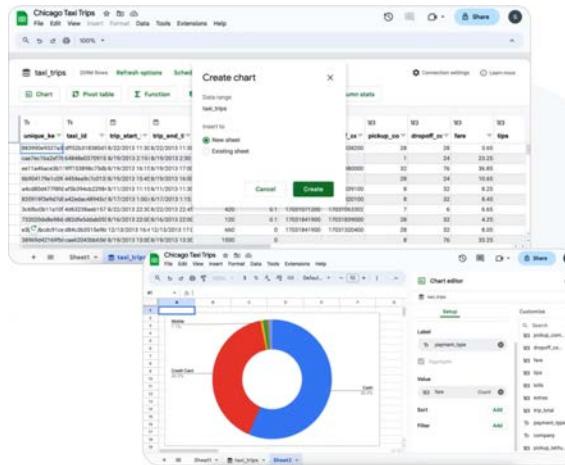
**JOB INFORMATION**

| Row | geo_id      |
|-----|-------------|
| 1   | 72031050402 |
| 2   | 72061040421 |

**Explore with Sheets** Analyze big data with a live connection in a familiar spreadsheet tool.

**Explore with Looker Studio** Visualize results and create live dashboards from your data.

## 2. Data Analysis Tools in Sheets:



- Charts:** Create visuals, like pie charts, for data analysis.
- Functions:** Use familiar spreadsheet functions to manipulate data.
- Pivot Tables:** Analyze data quickly with pivot tables.
- Data Extracts:** Filter and extract specific data points for more focused analysis.

## 3. Data and Refresh Options:

- Data connected to BigQuery remains live, meaning updates in BigQuery can be reflected in your Sheet.
- **Manual Refresh:** Users can manually refresh individual objects or the entire sheet.
- **Scheduled Refreshes:** Automatic refreshes can be set up with a maximum frequency of once per hour, ensuring data stays up-to-date.

#### 4. Security and Performance:

- Data isn't stored locally on individual machines, reducing security risks and preventing accidental manipulation.
- Connected Sheets avoids the performance limitations of traditional spreadsheets, enabling analysis on massive datasets without risking corrupt files or slow load times.

## How to Differentiate Export Options:

- **Live Connection via Connected Sheets:** Allows continuous, real-time interaction with the data.
- **Static Export via BigQuery:** Exports the results of a query without a live connection, functioning like a simple copy-paste into the sheet.

## ▼ Common Data Visualization Pitfalls

In data visualization, several common pitfalls can impact the effectiveness, accuracy, and performance of dashboards and reports. Let's dive into one major pitfall and the strategies for avoiding it.

### Data Freshness vs. Performance

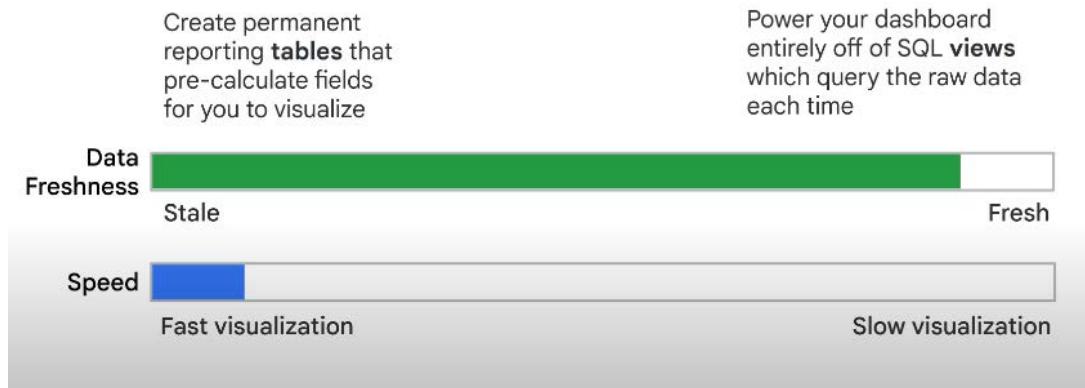
Where is your dashboard pulling data from?



A frequent challenge in data visualization is balancing data freshness with dashboard performance. This issue often arises when analysts decide how to retrieve and present the data in their dashboards.

### Precomputed Data vs. Real-Time Queries

When designing dashboards, you may need to pull data from various sources, which raises an important question: *How up-to-date does the data need to be?*



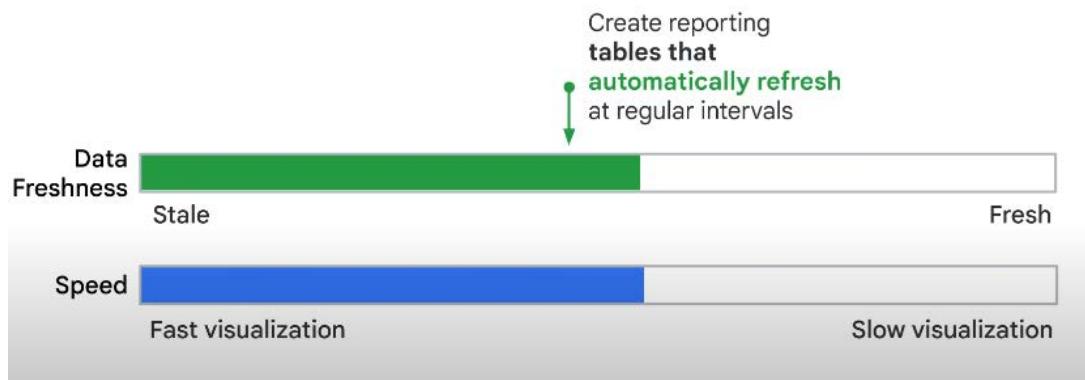
### 1. Precomputed Data (Table):

- **Benefits:** Precomputed tables hold processed data, which allows dashboards to load quickly since they are only retrieving and displaying results that have already been computed.
- **Pitfall:** Data can become stale if not refreshed frequently, leading to outdated insights.

### 2. Real-Time Queries (Views):

- **Benefits:** Running queries on the fly ensures you're always using the most current data in your visualizations.
- **Pitfall:** This approach can significantly slow down dashboards, as complex queries must run every time the dashboard loads. This can affect performance, especially with large datasets.

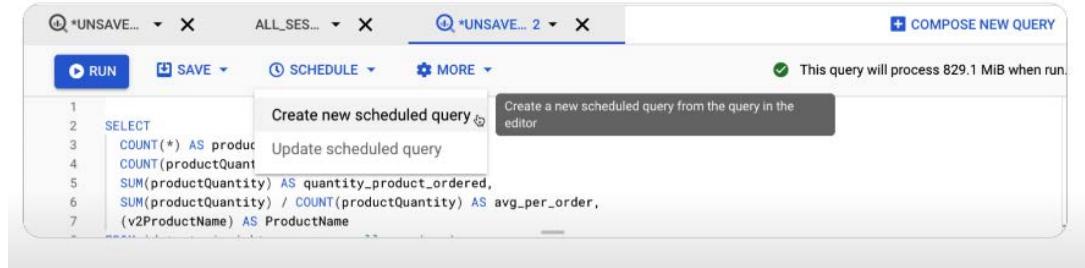
## Recommendation: Find a Balance



A good middle ground is using **scheduled queries** to create *reporting tables* that refresh periodically. This approach:

- Ensures your data remains relatively fresh.
- Reduces the performance load caused by running complex queries in real-time.
- Allows you to update data at intervals (e.g., hourly, daily) to strike a balance between real-time data accuracy and performance optimization.

## Scheduled Queries



The screenshot shows a query editor interface with three tabs at the top: \*UNSAVE... (active), ALL\_SES... (disabled), and \*UNSAVE... 2 (disabled). Below the tabs are buttons for RUN, SAVE, SCHEDULE, and MORE. A status message indicates: "This query will process 829.1 MiB when run." A tooltip over the SCHEDULE button says: "Create new scheduled query" and "Create a new scheduled query from the query in the editor". The main area contains a SQL query:

```

1 SELECT
2   COUNT(*) AS productCount,
3   COUNT(productQuantity) AS quantity_product_ordered,
4   SUM(productQuantity) / COUNT(productQuantity) AS avg_per_order,
5   (v2ProductName) AS ProductName
6
7

```

Scheduled queries can automate data updates without the need for manual intervention. They can be set up to run on a schedule (e.g., daily or weekly) and help maintain data freshness without running queries on the fly. These queries can include both **DDL (Data Definition Language)** and **DML (Data Manipulation Language)** statements.

New scheduled query

Details and schedule

Name for scheduled query

Schedule options

Start now     Schedule start time

Repeats                      Start date and run time

Daily                      4/22/19, 2:53 PM PDT

**⚠ This schedule will run Every day at 14:53 America/Los Angeles**

Destination for query results

**ⓘ A destination table is required to save scheduled query options.**

Project name              Dataset name

data-to-insights            ecommerce

Table name

Letters, numbers, and underscores allowed

Destination table write preference

Append to table     Overwrite table

Notification options

Send email notifications

Additionally, **parameterizing** the queries can further optimize results by storing data based on time and date parameters, ensuring more flexibility in organizing your datasets.

## BigQuery BI Engine: Improving Performance

Always **fresh**  
Always **fast**

**Democratize BI** by  
enabling data and business  
analysts to perform  
interactive analytics in **real  
time** at scale

For real-time dashboards, **BigQuery BI Engine** can help overcome performance challenges. BI Engine is an in-memory analysis service that accelerates query

performance and reduces latency, allowing for sub-second query times without the need to prebuild OLAP cubes or complex structures.

### Key Benefits of BigQuery BI Engine:



- **Sub-Second Query Latency:** Speeds up real-time analytics dashboards by caching frequently accessed data, providing near-instant query results.
- **Ease of Use:** No need for manually building and managing OLAP cubes.
- **In-Memory Intelligence:** Data is stored and processed in memory, reducing the time required to access large datasets.

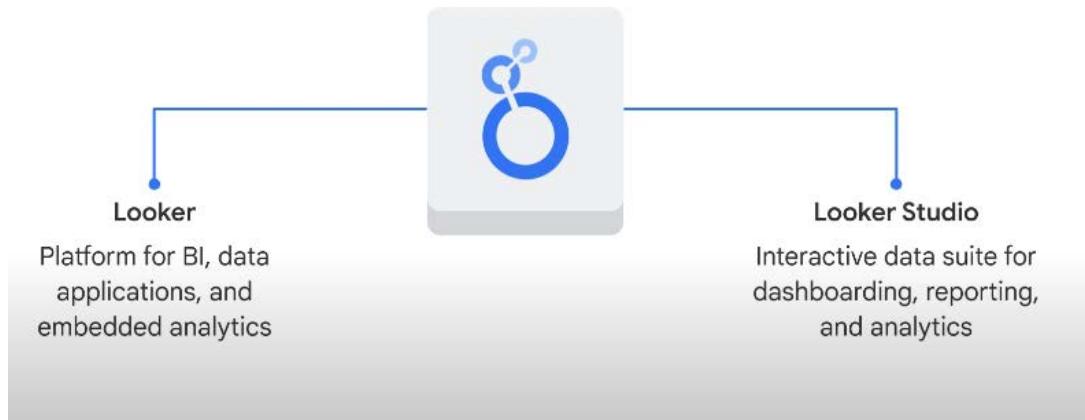
## Conclusion

When designing data visualizations, be mindful of the **data freshness vs. performance trade-off**. Scheduled queries and tools like **BigQuery BI Engine** can help optimize this balance, ensuring that you provide fast, up-to-date insights to your audience without compromising dashboard performance.

## ▼ Looker Studio

**Looker Studio**, formerly known as **Google Data Studio**, is a web-based tool that simplifies the creation of interactive dashboards and reports. This tool allows users to pull data from a wide variety of sources, including Google connectors and third-party data sources, making it an accessible platform for generating data visualizations and driving smarter business decisions.

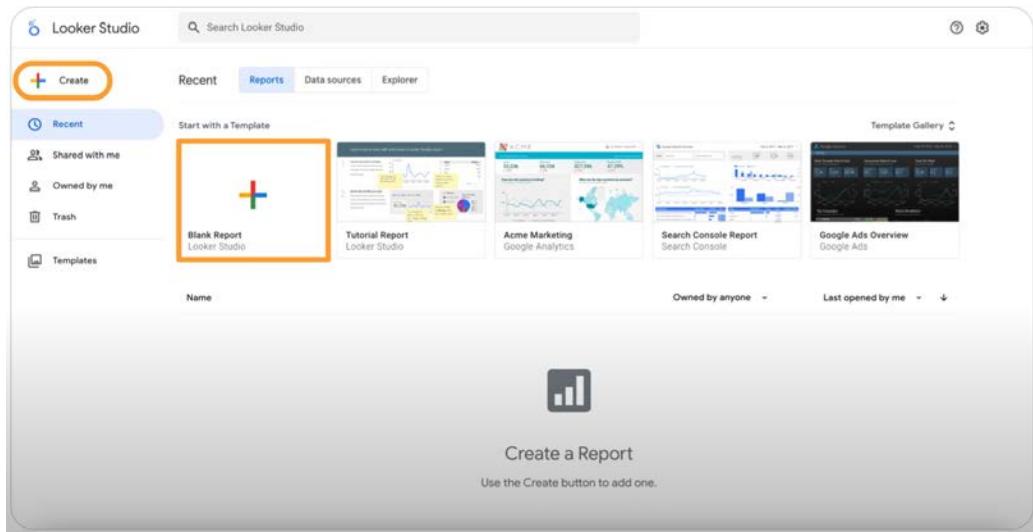
## Looker vs. Looker Studio



- **Looker** is a powerful business intelligence platform designed for data exploration and analytics at scale.
- **Looker Studio** is a more user-friendly, accessible tool aimed at creating interactive dashboards and reports using data from various sources.

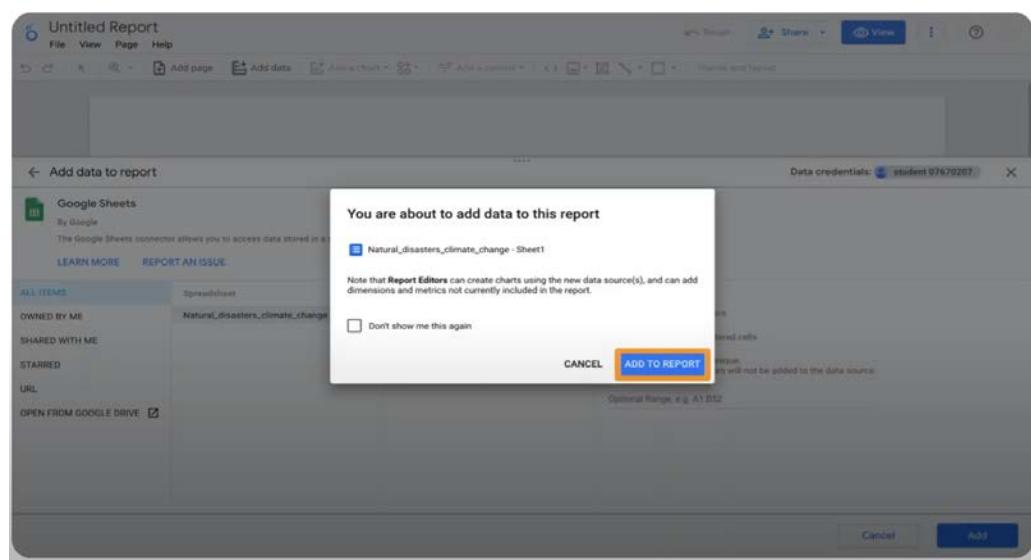
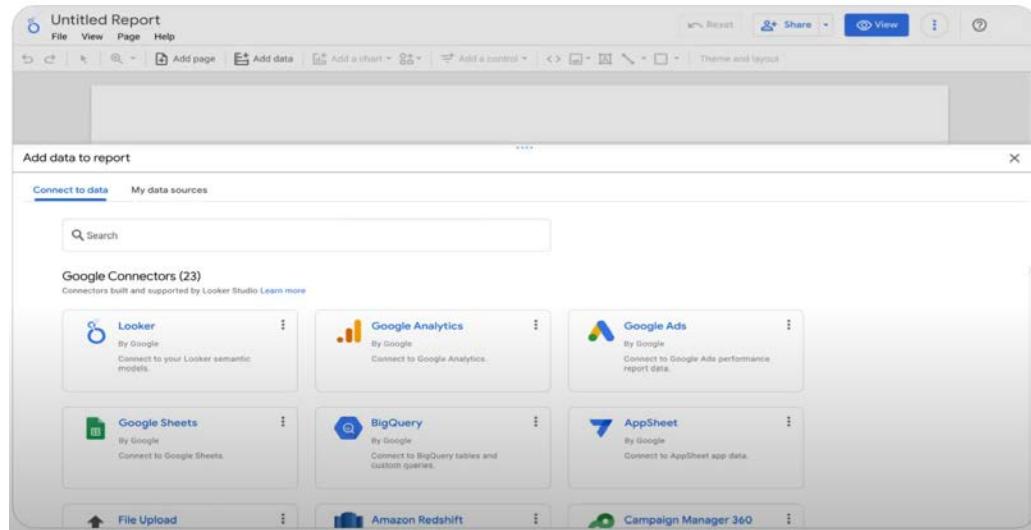
## Key Features of Looker Studio

### 1. Creating a Report



- Users can start by selecting a **Blank Report** from the template gallery or by using the **Create** button.
- Multiple data sources can be added to a single report, but sharing the report may allow others to view the data, so users need to be cautious when adding sensitive data sources.

### 2. Data Sources



- **Google Connectors:** Connect to Google Sheets, BigQuery, Google Ads, Google Analytics, and more.
- **Partner Connectors:** Access an expanding list of third-party sources through partner connectors.
- Data sources act as a pipeline to fetch data, which can be used across multiple reports and dashboards.

### 3. Dimensions and Metrics

Untitled Report

Data Last Updated: 3/1/2023 11:21:07 AM

**Chart**

**SETUP**

Data source: Natural\_disasters\_climate\_change...

BLEND DATA

Data Range Dimensions: Year

Dimensions: Year

Metric: Earthquake, Epidemic, Storm, Wildfire, Volcanic activity, Insect infestation, Extreme temperature, Landslide, Mass movement (dry), Flood, Drought

Optional metrics: Record Count

**Data**

Search: Natural\_disasters\_climate\_change...

Drought, Earthquake, Epidemic, Storm, Wildfire, Volcanic activity, Insect infestation, Extreme temperature, Landslide, Mass movement (dry), Flood, Drought

**Properties**

01 Data

02 Dimensions

03 Metrics

Untitled Report

Natural\_disasters\_climate\_change - Sheet...

Data credentials: student 07670207

Data freshness: 15 minutes

Community visualizations access: On

**EDIT CONNECTION** | **FILTER BY EMAIL**

**ADD A FIELD** **ADD A PARAMETER**

**Properties**

| Field               | Type        | Default Aggregation | Description |
|---------------------|-------------|---------------------|-------------|
| Insect infestation  | 123 Number  | Sum                 |             |
| Landslide           | 123 Number  | Sum                 |             |
| Mass movement (dry) | 123 Number  | Sum                 |             |
| Storm               | 123 Number  | Sum                 |             |
| Volcanic activity   | 123 Number  | Sum                 |             |
| Wildfire            | 123 Number  | Sum                 |             |
| Year                | Year (YYYY) | None                |             |
| Record Count        | 123 Number  | Auto                |             |

REFRESH FIELDS

13 / 13 Fields

- Dimensions:** These represent qualitative data (e.g., categories, dates).
- Metrics:** Also known as measures in other visualization tools, metrics represent quantitative data (e.g., sums, averages).
- These can be selected from the data source and used to create visualizations such as tables, charts, and more.

#### 4. Editing and Customizing Data

You can edit data sources or create **calculated fields** to customize your report. For example, the **CASE statement** defines conditions and returns results when those conditions are met.

CASE

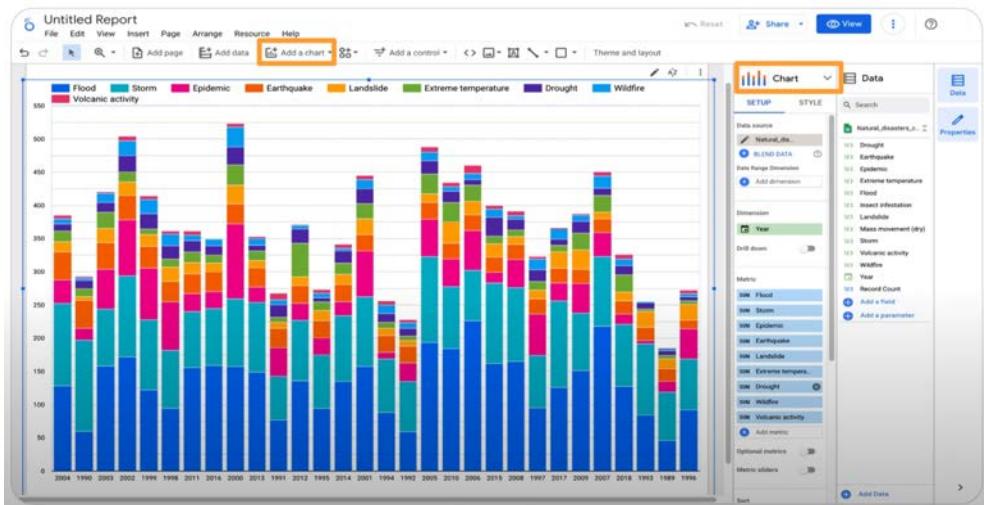
```
WHEN Country IN ("USA", "Canada", "Mexico") THEN "North America"  
WHEN Country IN ("England", "France") THEN "Europe"  
ELSE "Other"  
END
```

Additionally, you can:

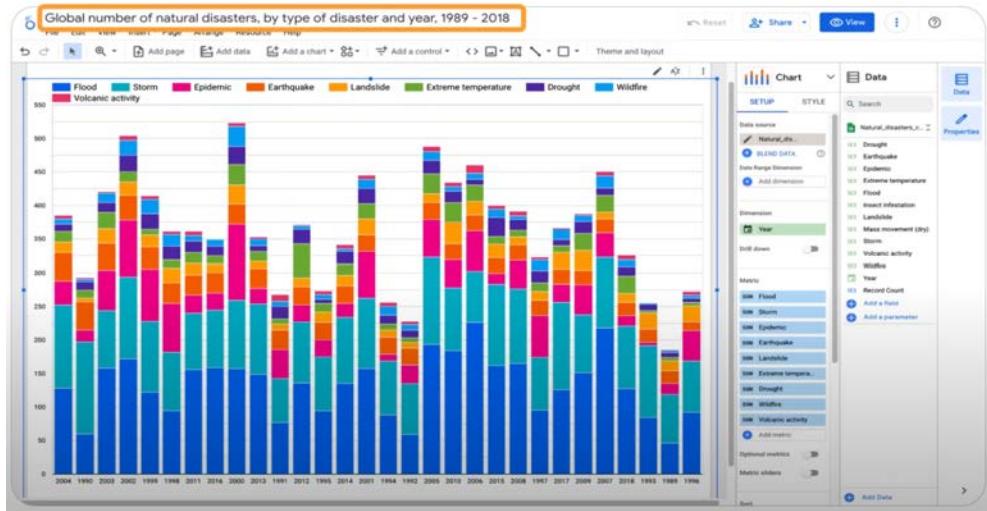
- Change the data table view to a chart.
- Edit the chart style or revert to the data table view.
- Add separate charts or resize components on the canvas.

## 5. Chart Customization

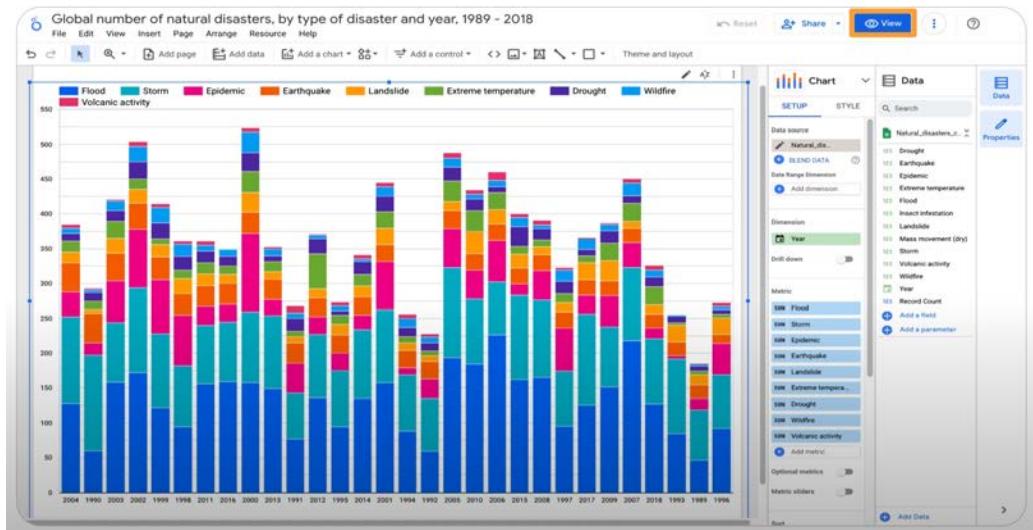
- Looker Studio provides flexibility in the visualization process, allowing users to switch between chart types (e.g., bar charts, pie charts) and customize chart styles.



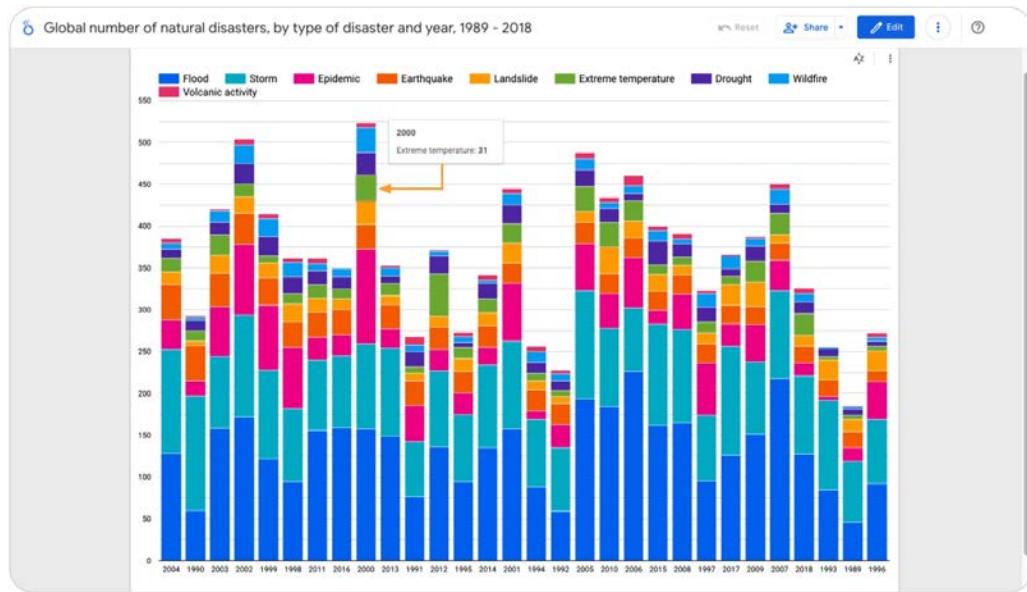
- Users can add charts from the toolbar, resize components, and drag fields to change their order in the display.



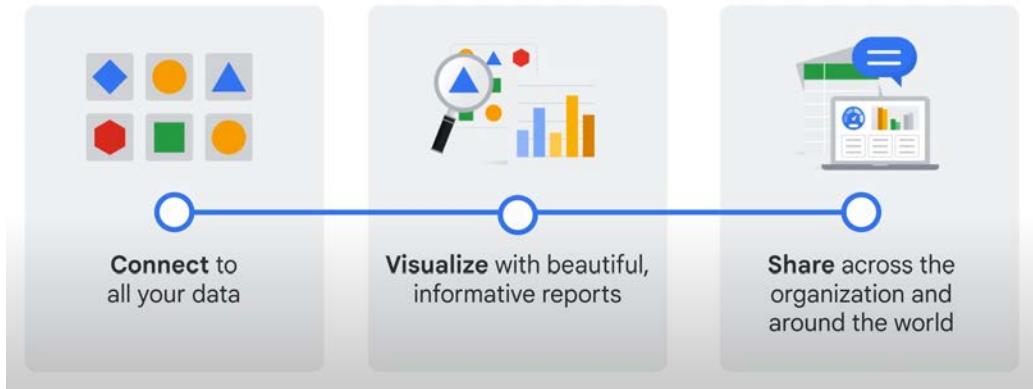
## 6. Interactivity and Viewing



- Reports and dashboards in Looker Studio are interactive. For example, viewers can hover over charts to see live data.
- Viewers can only interact with, but not modify, the reports unless given permission to edit.



## 7. Sharing and Collaboration



- Sharing in Looker Studio is similar to sharing in **Google Drive**. Users can share reports with others, but sharing a report does not grant direct access to the underlying data sources. Data sources must be shared separately.
- Looker Studio enables collaboration while keeping sensitive data secure.

## ▼ 6- Developing scalable data transformations pipelines in BigQuery with Dataform

### ▼ What is Dataform?

Dataform is a tool that helps data teams build, manage version control, and orchestrate SQL pipelines specifically in BigQuery. It enables teams to create efficient workflows, improve data quality, and centralize data for analytics.

## Shift from ETL to ELT



Organizations are gradually moving away from traditional **ETL** (Extract, Transform, Load) processes, which rely on dedicated ETL tools and environments like Spark, to a more modern **ELT** (Extract, Load, Transform) approach. With ELT, SQL pipelines are built directly within BigQuery.

### Why the Shift?

- 1. Serverless Architecture:** BigQuery is serverless, so companies don't need to manage infrastructure like Spark clusters.
- 2. Skill Availability:** There is a larger pool of analysts skilled in SQL compared to those proficient in Spark or other ETL tools.

## Centralizing Raw Data in BigQuery



### Challenges with Raw Data

Customers who adopt the ELT approach centralize raw data from various systems into BigQuery. This results in **thousands of incompatible data tables** that must be transformed before analytics can take place.

### Transformation Layers

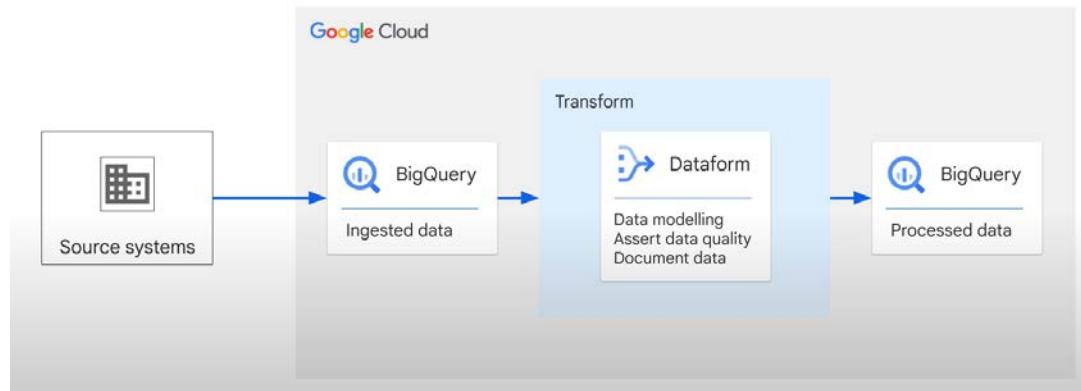
To address these challenges, data is processed through several transformation layers, which refine the raw data, ultimately leading to a **single source of truth**.

This layer has the necessary **quality, completeness, and freshness** to support analytics, reporting, and decision-making.

## Common Challenges in Data Transformation

1. **Manual Processes:** Many steps in data transformation are manual and distributed across various tools and processes.
2. **Lack of Reusability:** SQL statements cannot be easily reused across different scripts.
3. **No Testing:** There's no built-in way to write tests for ensuring data consistency.
4. **Dependency Management:** Managing dependencies between processes requires external software solutions.
5. **Metadata Management:** Documentation and metadata are often treated as afterthoughts and need to be maintained in an external catalog.

## How Dataform Solves These Challenges



### Scalable Pipelines with SQL

Dataform allows teams to develop and operate scalable data transformation pipelines directly in BigQuery using SQL. The framework enables the creation of a **central data model**—a single source of truth where tables are:

- Curated
- Continuously updated
- Documented
- Tested

## Key Features

Dataform's framework lets teams:

- Define tables
- Ensure data quality with assertions
- Document tables

All of this can be done in a single environment without the need for managing infrastructure or external dependencies.

---

## Collaborative Workflows and Best Practices

Dataform also promotes collaboration through best practices commonly found in software development, such as:

- **Version Control**
- **Environments**
- **CI/CD (Continuous Integration/Continuous Deployment)**
- **Testing**
- **Documentation**

The tool empowers data analysts to build **production-grade SQL pipelines**, eliminating the need for dedicated data engineers.

---

## Open-Source Dataform Core

Dataform core is part of the open-source Dataform data modeling framework, which includes the **Dataform CLI**. Using the CLI, teams can compile and run Dataform core locally, and deploy assets to various data warehouses, including:

- BigQuery
- Snowflake
- Redshift
- Azure SQL Data Warehouse
- Postgres

Since it's open source, Dataform core can also be extended with custom integrations.

---

## What is SQLX?

SQLX is an **open-source extension of SQL**, and it's the primary tool used in Dataform. Since it extends SQL, any SQL file is also a valid SQLX file.

### Standard SQL

```
CREATE OR REPLACE TABLE
  'projectid.dataset.new_table' AS
SELECT
  country AS country,
  CASE
    WHEN country IN ('US', 'CA') THEN 'NA'
    WHEN country IN ('GB', 'FR', 'DE') THEN 'EU'
    WHEN country IN ('AU') THEN country
    ELSE 'Other countries'
  END
  AS country_group,
  device_type AS device_type,
  SUM(revenue) AS revenue,
  SUM(pageviews) AS pageviews
FROM
  'projectid.dataset.table_1'
GROUP BY
  1,2,3
```



### SQLX file

```
config { type: "table" }

SELECT
  country AS country,
  ${mapping.region("country")} AS country_group,
  device_type AS device_type,
  SUM(revenue) AS revenue,
  SUM(pageviews) AS pageviews
FROM
  ${ref("dataset_1")}
GROUP BY
  1,2,3
```

## Additional Features of SQLX

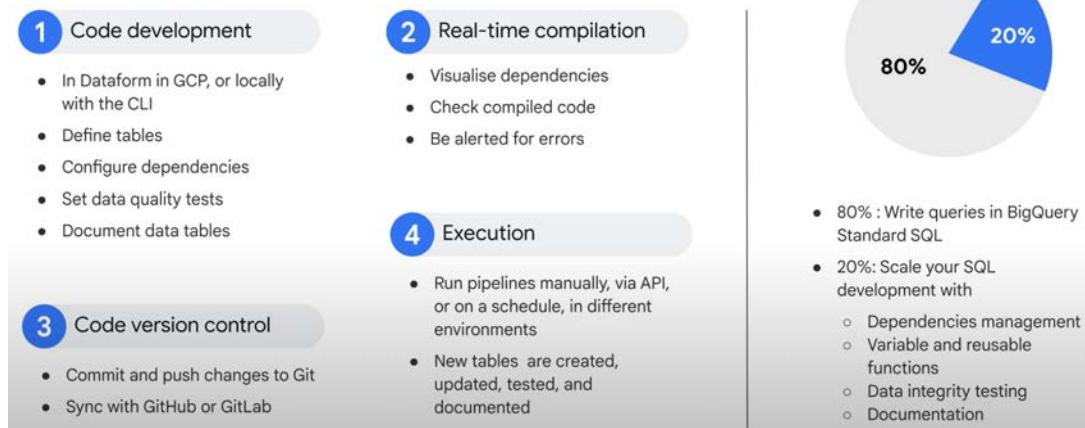
SQLX adds features to make development faster, more reliable, and scalable, such as:

- **Dependency Management**
- **Automated Data Quality Testing**
- **Data Documentation**

## ▼ Getting Started with Dataform

In this lesson, we will cover the basics of **getting started with Dataform**. By the end of the module, you'll gain hands-on experience through a lab session. The topics will include the **Dataform code lifecycle** and step-by-step guidance on setting up your Dataform environment.

## Code Lifecycle in Dataform



### 1. Raw Data to Transformation:

- After raw data is extracted from source systems and loaded into **BigQuery**, Dataform helps transform it into well-defined, tested, and documented data tables through code.

### 2. Real-Time Compilation:

- Dataform compiles the SQL workflow code in your workspace to **SQL** in real time, creating a compilation result that can be executed directly in BigQuery.

### 3. Dataform.json File:

- The compilation result is generated using settings defined in the `dataform.json` file, where you can control how and where Dataform executes your SQL workflows.

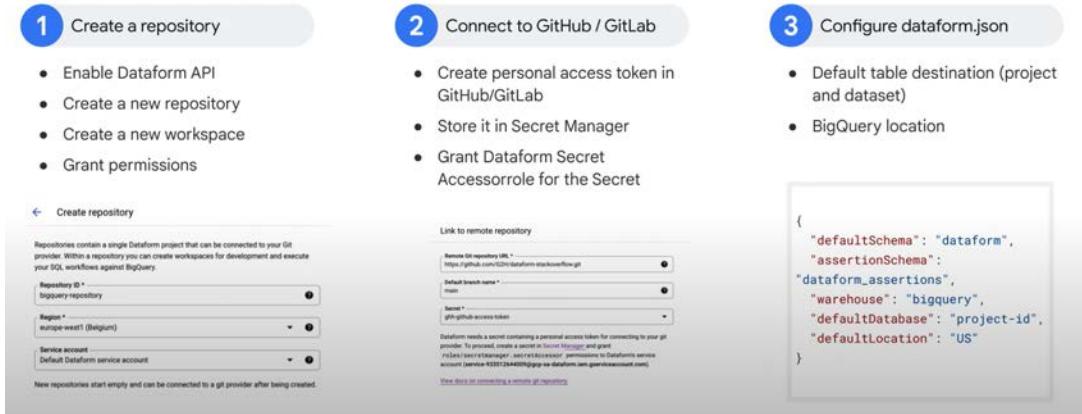
### 4. Git Integration:

- Dataform integrates with multiple **Git providers**, enabling collaboration between data analysts and engineers on the same repository.

### 5. Customizing Executions:

- You can tailor the compilation result, manually trigger executions, or schedule them to control when the SQL workflows or selected elements are executed.

## Setting Up Dataform



## 1. Create a Repository:

- The first step is setting up a **repository**. Each repository contains a single Dataform project connected to a Git provider.
- The repository houses a collection of **SQLX**, **JavaScript** files, Dataform configuration files, and packages that make up the SQL workflow.

## 2. Development Workspace:

- A **development workspace** is an editable copy of the repository where you can create, edit, or delete content without affecting others.
- After making changes, you can **commit** and **push** them to the repository.

## 3. Developing SQL Workflows:

- In the workspace, you can develop SQL workflows using Dataform core with **SQLX** and JavaScript, or purely with JavaScript.

## 4. Linking to Git:

- Dataform repositories can be linked to remote Git repositories hosted by providers like **GitHub**, **GitLab**, and **Bitbucket Cloud**. This allows you to **push** and **pull** changes between Dataform and the remote repository.

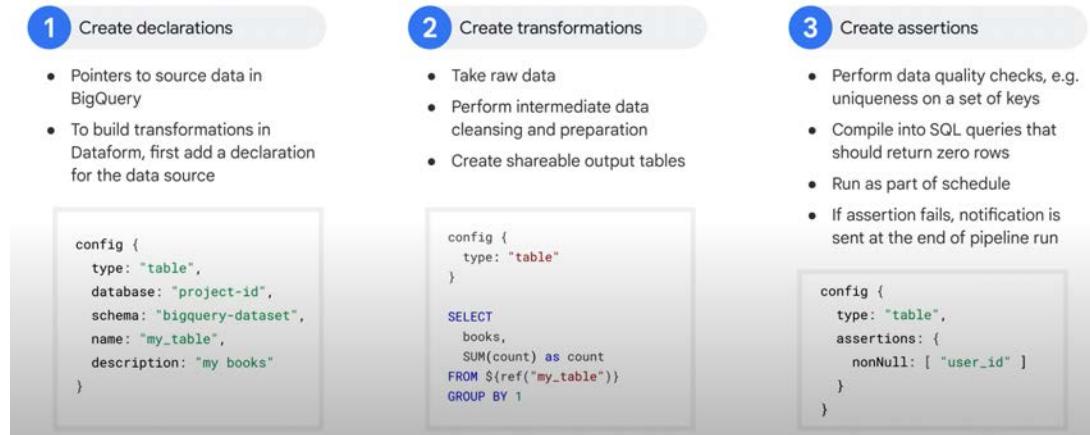
## 5. Configuring Dataform.json:

- Each repository has a `dataform.json` file that configures basic project settings like project ID, table prefixes, and schema suffix defaults.

## 6. BigQuery Permissions:

- You need to grant appropriate **IAM roles** and permissions to ensure your Dataform code can access **BigQuery** datasets. You'll configure these permissions in the lab at the end of the module.

# Developing and Executing Pipelines



## 1. Declarations:

- You can declare any **BigQuery table** as a data source in Dataform. These declarations allow you to add descriptions and tags, which are visible in BigQuery.
- You can also declare **dependencies** to define relationships between SQL workflow objects, ensuring that the execution of dependent objects relies on their dependencies.

## 2. Writing Transformations:

- Building transformations in Dataform involves writing SQL, leveraging previous SQL discussions to create data transformations efficiently.

## 3. Assertions (Data Quality Tests):

- An assertion is a **data quality test** that identifies rows violating specified rules. If any rows fail the test, the assertion fails.
- Dataform runs assertions automatically when updating the SQL workflow, notifying you if any fail.

## 4. Monitoring Workflow Executions:

The screenshot shows a Dataform execution status page. The top section displays execution details:

| Status         | Failed                   |
|----------------|--------------------------|
| Start time     | Aug 31, 2023, 8:58:52 AM |
| Duration       | <1s                      |
| Executed as    |                          |
| Failure reason | Access Denied:           |

The bottom section shows the executed SQL query:

```

1 BEGIN
2   CREATE SCHEMA IF NOT EXISTS: [REDACTED].dataform` OPTIONS(location='E
3 EXCEPTION WHEN ERROR THEN
4   IF NOT CONTAINS_SUBSTR(@@error.message, 'already exists: dataset') AND
5   NOT CONTAINS_SUBSTR(@@error.message, 'too many dataset metadata up
6   NOT CONTAINS_SUBSTR(@@error.message, 'User does not have bigquery.
7   THEN
8     RAISE USING MESSAGE = @@error.message;
9   END IF;
10 END;
11 BEGIN
12   DECLARE dataform_table_type DEFAULT (
13     SELECT ANY_VALUE(table_type)
14     FROM [REDACTED].dataform.INFORMATION_SCHEMA.TABLES
15     WHERE table_name = 'first_view'
16   );
17   IF dataform_table_type IS NOT NULL AND dataform_table_type != 'V
18   IF dataform_table_type = 'BASE TABLE' THEN
19     DROP TABLE IF EXISTS [REDACTED].dataform.first_view;
20   ELSEIF dataform_table_type = 'VIEW' THEN
21     DROP VIEW IF EXISTS [REDACTED].dataform.first_view;
22   ELSEIF dataform_table_type = 'MATERIALIZED VIEW' THEN
23     DROP MATERIALIZED VIEW IF EXISTS [REDACTED].dataform.first_view;
24   END IF;
25 END IF;
26 BEGIN
27   CREATE OR REPLACE VIEW [REDACTED].dataform.first_view
28   OPTIONS()
29   AS (
30

```

- Once your workflows are ready, use the **Executions tab** in the workspace to monitor their progress. This section allows you to track successful or failed executions.

Example: A failed execution may occur if the service account lacks the necessary privileges. You can view detailed error messages and troubleshoot accordingly.

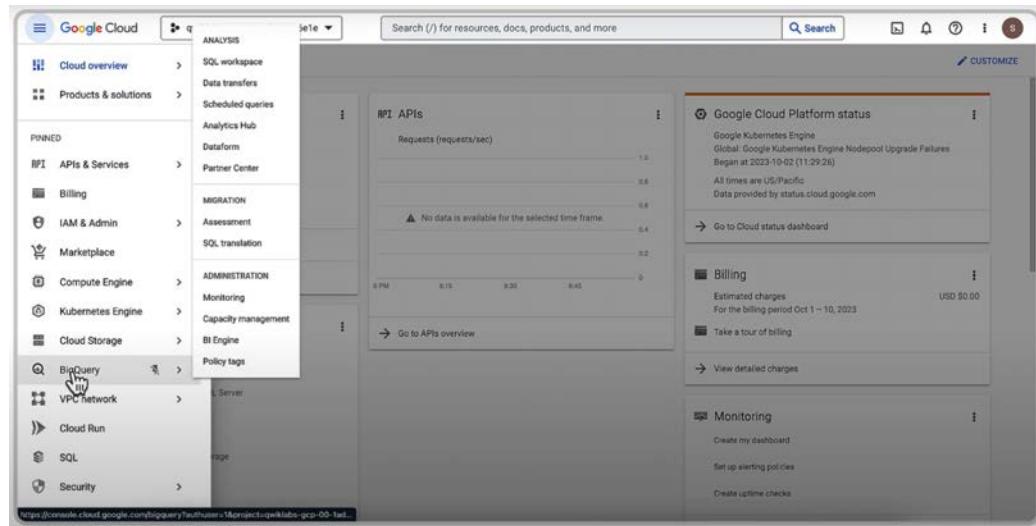
By setting up a repository, working in a development workspace, and defining dependencies and transformations, you'll be ready to start executing SQL workflows in Dataform. Ensure permissions and configurations are in place to avoid issues during execution, and use assertions to maintain data quality.

## ▼ Demo: Getting Started with Dataform

This demo will guide you through setting up Dataform, creating a repository, and initializing a development workspace.

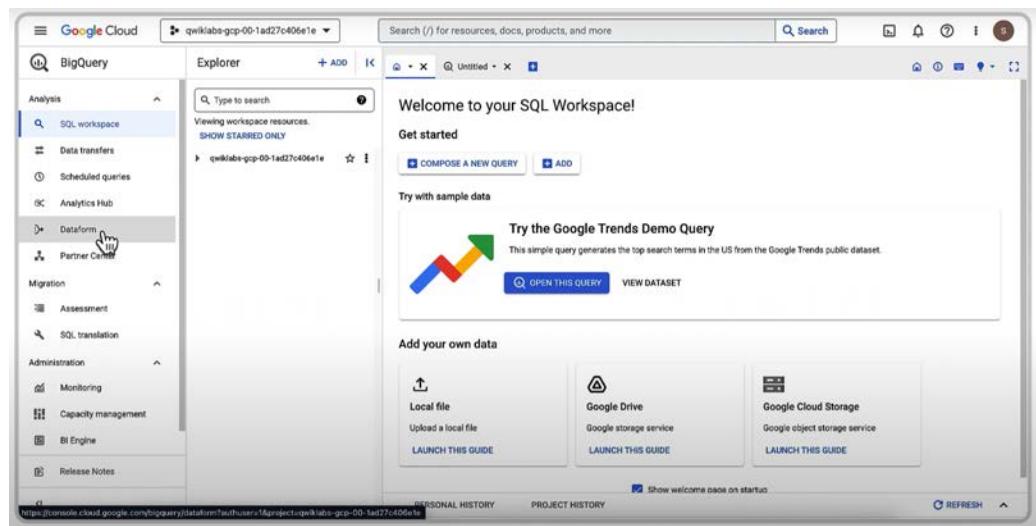
### Step 1: Access Google Cloud Console

- Start by logging into the Google Cloud Console.
- Navigate to **BigQuery** using the menu.

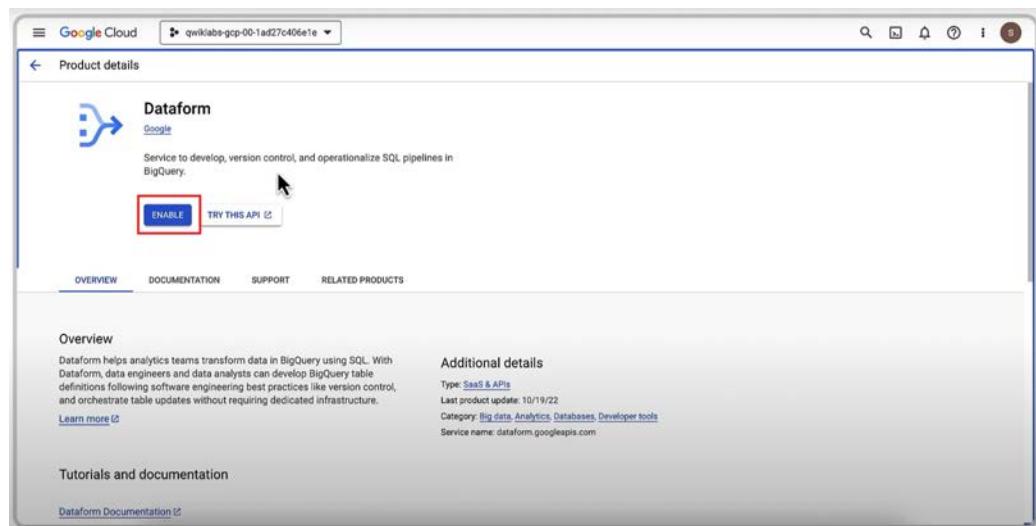


## Step 2: Enable Dataform

- On the left-hand side menu in BigQuery, click **Dataform**.

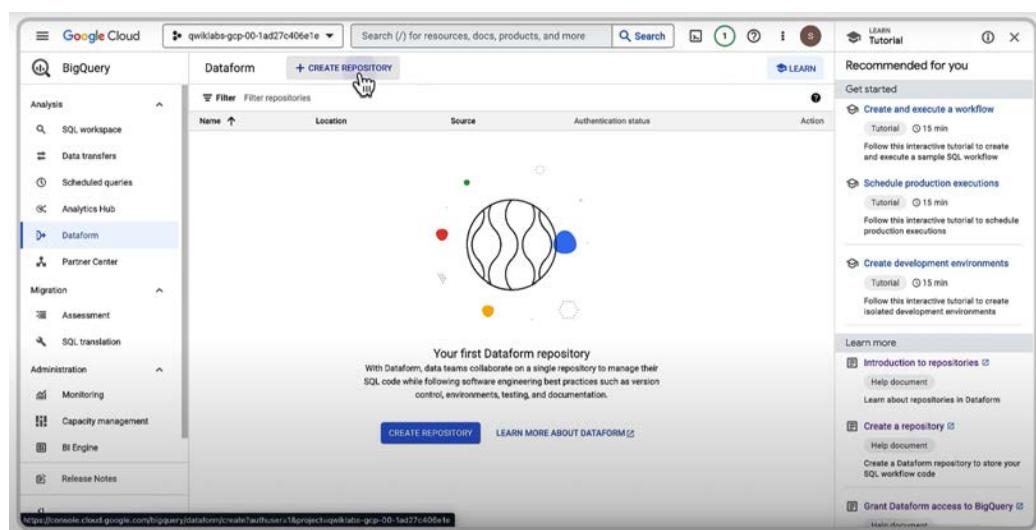


- Enable the Dataform service to proceed. You will be redirected to the Dataform user interface (UI).

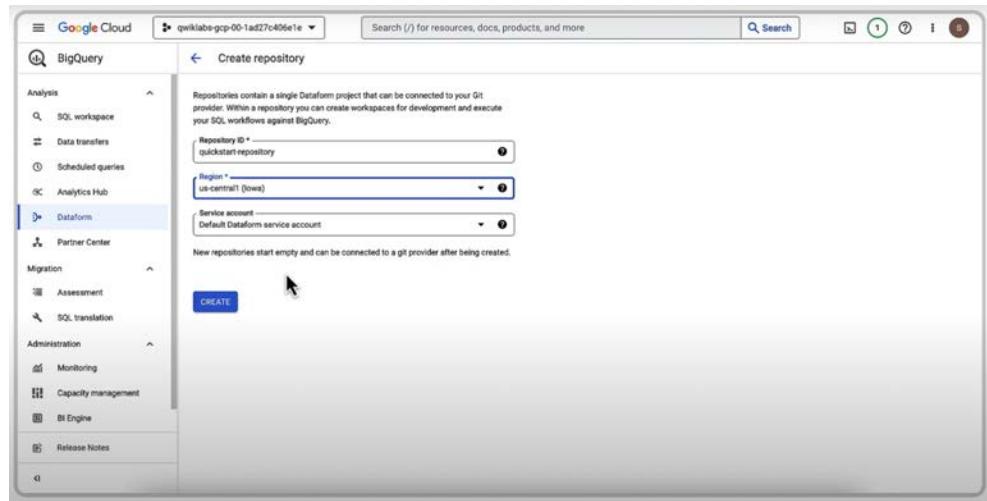


### Step 3: Create a Repository

- In the Dataform UI, create a new repository.



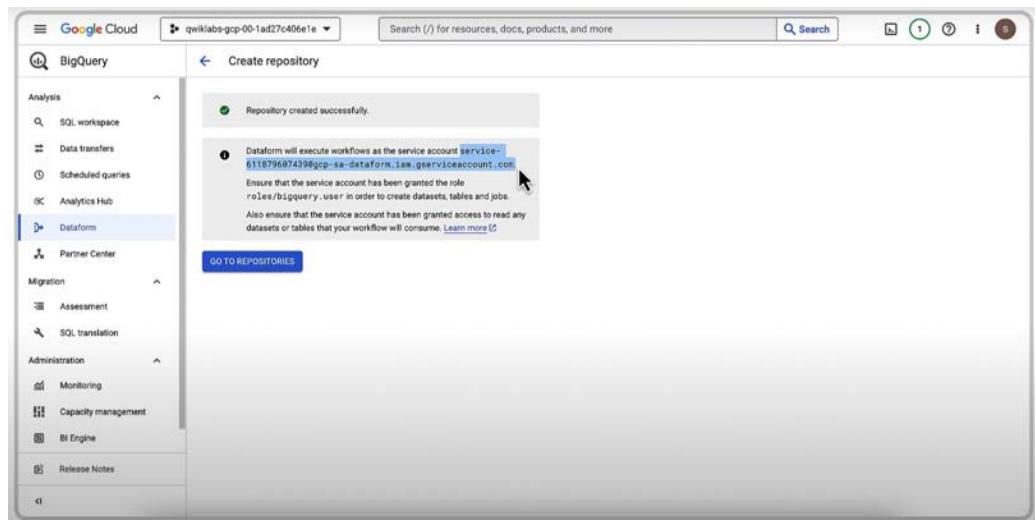
- A repository holds a single Dataform project, which can connect to your Git provider.
- Name the repository and choose a region. The default Dataform service account is pre-selected.



- Click **Create**.

#### Step 4: Set Up IAM Roles

- Copy the service account information.



- Navigate to **IAM and Admin > IAM** in the Google Cloud Console.
- Under "View By Principals," click **Grant Access** and paste the service account information.
- Assign the following IAM roles:
  1. BigQuery Job User
  2. BigQuery Data Editor
  3. BigQuery Data Viewer

#### Step 5: Create a Development Workspace

The screenshot shows the Google Cloud Dataform interface. On the left, there's a sidebar with various options like Analysis, Dataform, and Migration. The main area shows a table titled 'Dataform' with one row: 'quickstart-repository'. A cursor is hovering over the 'quickstart-repository' row. To the right, there's a 'LEARN' section with several tutorial cards: 'Get started' (Create and execute a workflow), 'Schedule production executions', 'Create development environments', and 'Learn more' sections for 'Introduction to repositories', 'Create a repository', and 'Grant Dataform access to BigQuery'.

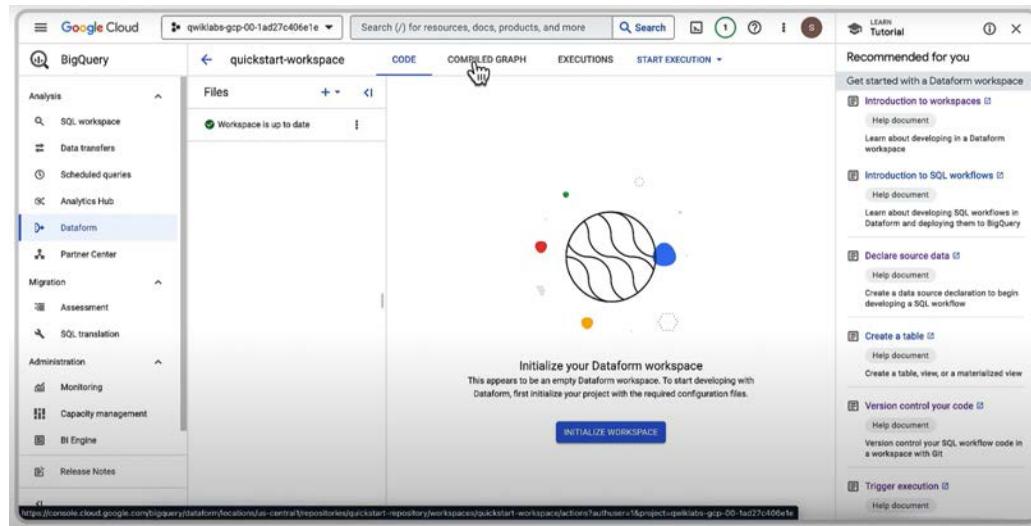
- Now, in the repository, create a **development workspace** where you will develop and run your SQL workflows.

The screenshot shows the 'Create development workspace' dialog. It has fields for 'Name' (set to 'quickstart-workspace') and 'Workspace ID' (also set to 'quickstart-workspace'). A red box highlights the 'CREATE' button, which is being clicked by a cursor. The background shows the Dataform interface with the 'quickstart-repository' selected.

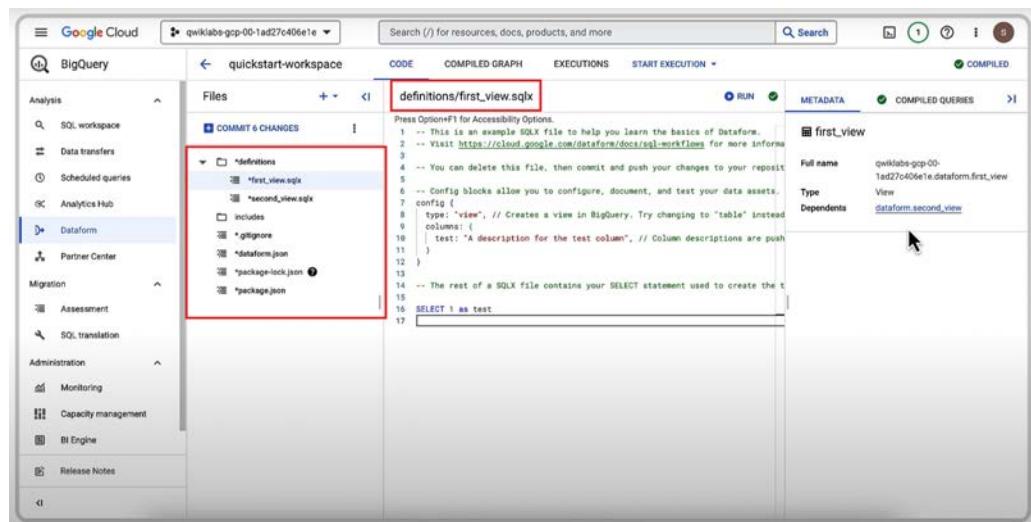
- Name the workspace and click **Create**.

## Step 6: Initialize Workspace

- In the workspace, you'll see several tabs: Code, Compiled Graph, Executions, and Start Execution.



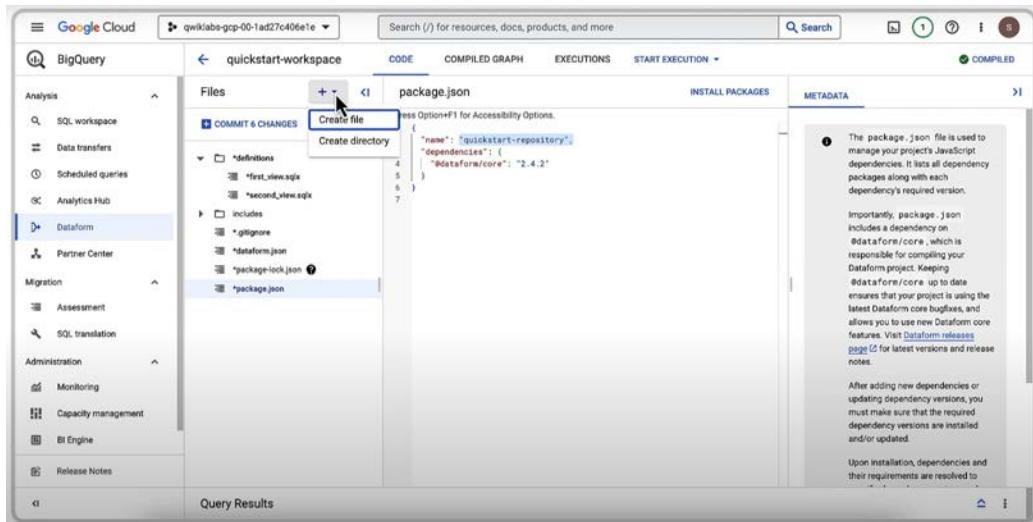
- Initialize the workspace to set up the file structure.
- Pre-populated starter files will appear in the **Files** pane.



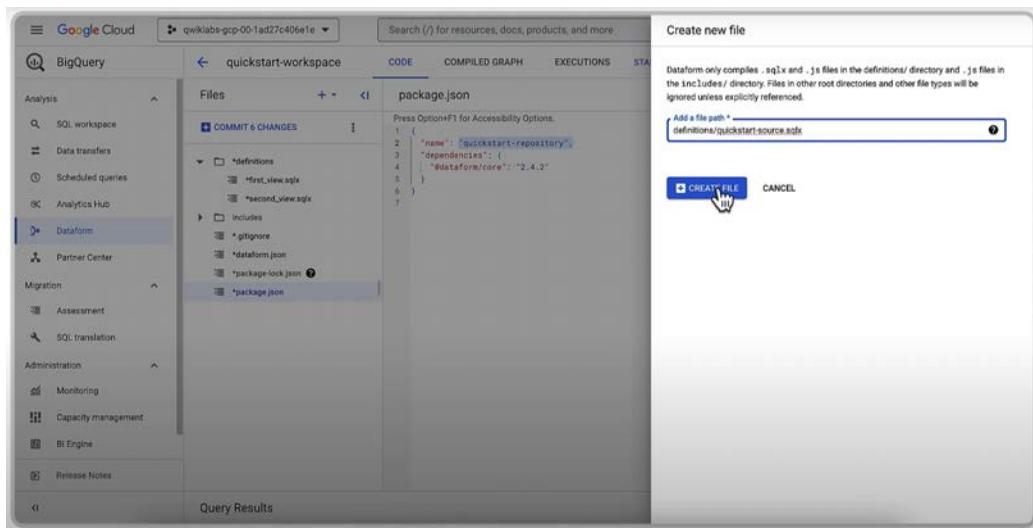
- The **definitions** folder is where you'll create your SQLX files, such as views or tables.

## Step 7: Add a New SQLX File

- Click the dropdown button and select **Create File** to add a new SQLX file.



- Name the file and click **Create File**.



- Write your SQL code in the editor. For this demo, you can write a simple **SELECT** statement to create a view.

```

1 config {
2   type: "view"
3 }
4 SELECT
5   "apples" AS fruit,
6   1 AS count
7 UNION ALL
8 SELECT
9   "oranges" AS fruit,
10  1 AS count
11 UNION ALL
12 SELECT
13   "pears" AS fruit,
14   1 AS count
15 UNION ALL
16 SELECT
17   "bananas" AS fruit,
18   0 AS count

```

- The code will compile immediately, showing the object type (view).

```

1 config {
2   type: "view"
3 }
4 SELECT
5   "apples" AS fruit,
6   1 AS count
7 UNION ALL
8 SELECT
9   "oranges" AS fruit,
10  1 AS count
11 UNION ALL
12 SELECT
13   "pears" AS fruit,
14   1 AS count
15 UNION ALL
16 SELECT
17   "bananas" AS fruit,
18   0 AS count

```

## Step 8: Add Another File for a Table

- Similarly, create another file for a table.
- The table code will reference the previously created view to populate rows.

The screenshot shows the Google Cloud BigQuery Dataform interface. The left sidebar is collapsed. The main area displays a file named 'definitions/quickstart-table.sqlx'. The code defines a table named 'quickstart-table' with a single column 'fruit' and a calculated column 'count'. The code is highlighted with syntax coloring. A red box highlights the file name 'quickstart-table.sqlx'. The right panel shows the 'METADATA' tab for the table, which includes the full name 'qwiklabs-gcp-00-1ad27c406e1e.dataform.quickstart-table' and type 'Operations'. The status bar at the bottom says 'Query Results'.

- The file will also compile, showing any relevant output or errors.

The screenshot shows the Google Cloud BigQuery Dataform interface. The left sidebar is collapsed. The main area displays a file named 'definitions/quickstart-table.sqlx'. The code is identical to the previous screenshot. However, the right panel shows an error message: 'Not found: Dataset qwiklabs-gcp-00-1ad27c406e1e.dataform was not found in location US.' A red box highlights this error message. The status bar at the bottom says 'Query Results'.

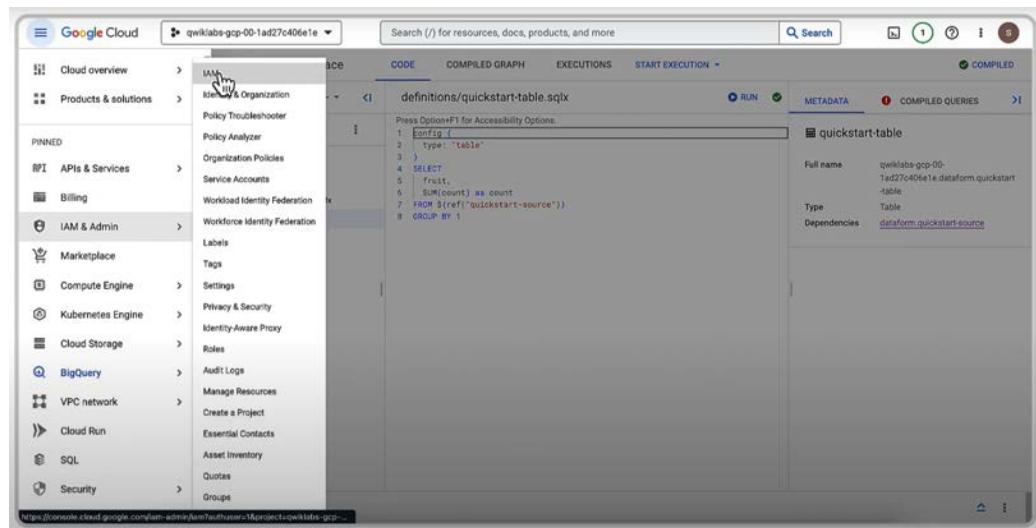
- Because type is different.

The screenshot shows the Google Cloud BigQuery Dataform interface. On the left, the navigation menu includes Analysis, SQL workspace, Data transfers, Scheduled queries, Analytics Hub, Dataform (selected), Partner Center, Migration, Assessment, SQL translation, and Administration. The main area displays a code editor with the file `definitions/quickstart-table.sqlx`. The code defines a table named `quickstart-table` with a single column `fruit` and a count aggregation. To the right, the `METADATA` panel shows the table's full name, type (Table), and dependencies.

```

1 config {
2   type: "table"
3 }
4 SELECT
5   fruit,
6   SUM(count) as count
7 FROM ${ref("quickstart-source")}
8 GROUP BY 1
  
```

- In the navigation menu, go to IAM and Admin, and then IAM.

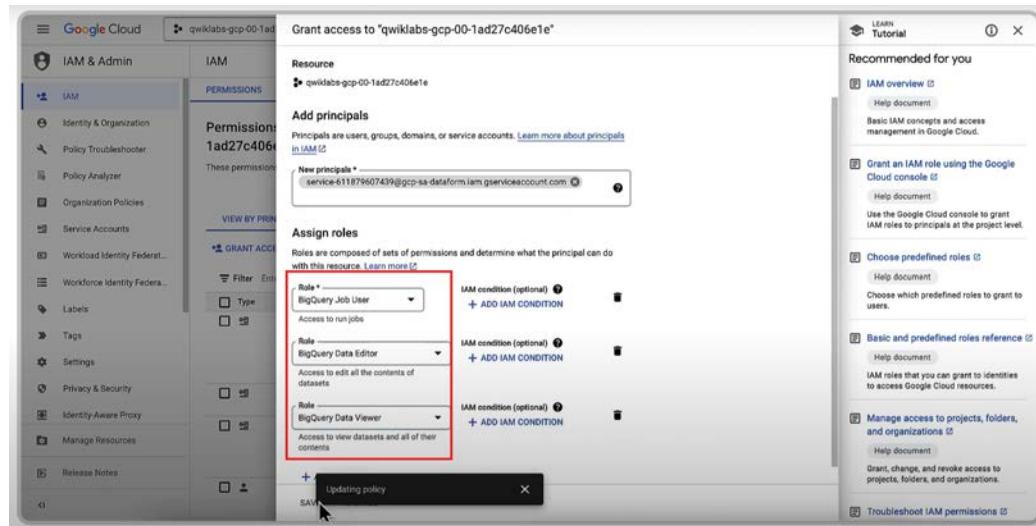


- Under View By Principals, click on Grant Access.

| Type   | Principal  | Name   | Role             | Security Insights | Inheritance |
|--|--|--|------------------|-------------------|-------------|
| compute@developer.gserviceaccount.com                | Compute Engine default service account               | 611879607439-compute@developer.gserviceaccount.com           | Editor           |                   |             |
| prod.iam.gserviceaccount.com                         | prod.iam.gserviceaccount.com                         | admin@qwiklabs-services-1ad27c406e1e.iam.gserviceaccount.com | Owner            |                   |             |
| qwiklabs-gcp-00-1ad27c406e1e.iam.gserviceaccount.com | qwiklabs-gcp-00-1ad27c406e1e.iam.gserviceaccount.com | qwiklabs-gcp-00-1ad27c406e1e.iam.gserviceaccount.com         | BIGQUERY ADMIN   |                   |             |
| student-03-6759fb1b7b6@qwiklabs.net                  | student-03-6759fb1b7b6@qwiklabs.net                  | student-03-6759fb1b7b6@qwiklabs.net                          | App Engine Admin |                   |             |
|  |  | 23dd682  | BIGQUERY ADMIN   |                   |             |

- This is where we paste the Dataform service account information we copied earlier when we created the Dataform repository.

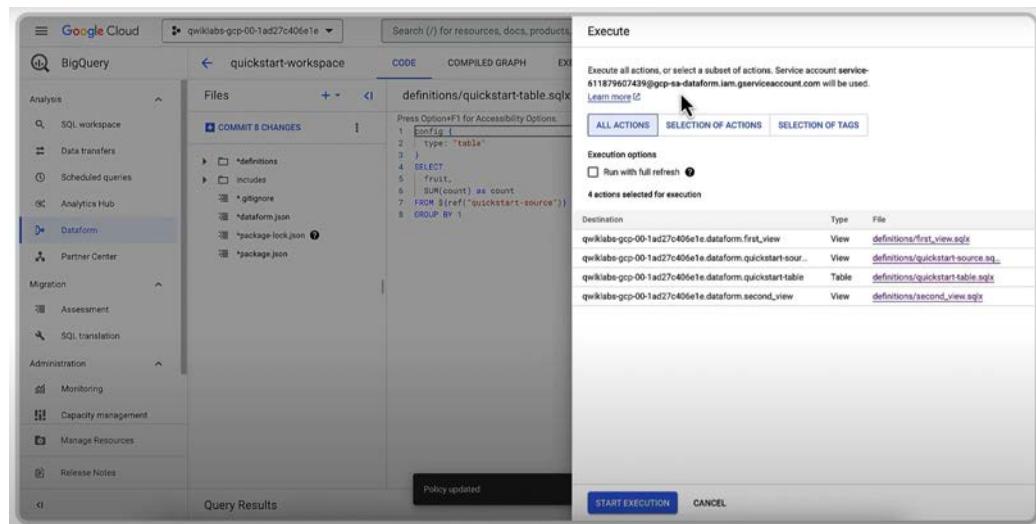
- And we essentially want to add 3 roles here.



- The first one is the bigquery job user.
- Next, we add the BigQuery Data Editor.
- And the final one is the BigQuery Data Viewer.

### Step 9: Execute Your Workflow

- Once that is added, we can now go back to the Dataform UI and navigate back to the repository, and then into the development workspace.
- Select **Start Execution**, then choose **All Executions**.
- Click **Start Execution** to run your SQL workflow



### Step 10: Review Execution Logs and Results

- Check the execution logs in the **Execution** tab or through the details link.

The screenshot shows the Google Cloud BigQuery Dataform interface. On the left, the navigation sidebar includes Analysis, Data transfers, Scheduled queries, Analytics Hub, Dataform (selected), Partner Center, Migration, Assessment, SQL translation, Administration (Monitoring, Capacity management, Manage Resources, Release Notes). The main area shows a file named 'definitions/quickstart-table.sqlx' with the following code:

```

1 config {
2   type: "table"
3 }
4 SELECT
5   fruit,
6   SUM(count) as count
7 FROM ${ref("quickstart-source")}
8 GROUP BY 1

```

Below the code, there are tabs for CODE, COMPILED GRAPH, EXECUTIONS, and START EXECUTION. A 'RUN' button is visible. To the right, there's a 'METADATA' section showing the full name as 'quickstart-table' and type as 'Table'. A dependency 'dataform.quickstart-source' is listed. At the bottom, a message says 'Successfully created workflow execution'.

- You will see the status of the run and details about each SQL file that was executed.

The screenshot shows the Google Cloud BigQuery Dataform interface with the title 'Oct 10, 2023, 10:13:20PM'. The left sidebar is identical to the previous screenshot. The main area displays 'Details' for the workflow run, including start time (Oct 10, 2023, 10:13:20PM), status (Success), duration (8 seconds), source type (Workspace), and source (quickstart-workspace). Below this is the 'Actions' section, which lists four events:

| Status  | Start time                | Duration  | Action            | Destination                      | Details                      |
|---------|---------------------------|-----------|-------------------|----------------------------------|------------------------------|
| Success | Oct 10, 2023, 10:13:21 PM | 2 seconds | quickstart-source | <a href="#">quickstart-table</a> | <a href="#">VIEW DETAILS</a> |
| Success | Oct 10, 2023, 10:13:21 PM | 3 seconds | first_view        | <a href="#">quickstart-table</a> | <a href="#">VIEW DETAILS</a> |
| Success | Oct 10, 2023, 10:13:25 PM | 2 seconds | quickstart-table  | <a href="#">quickstart-table</a> | <a href="#">VIEW DETAILS</a> |
| Success | Oct 10, 2023, 10:13:26 PM | 1 second  | second_view       | <a href="#">quickstart-table</a> | <a href="#">VIEW DETAILS</a> |

At the bottom, a message says 'Successfully created workflow execution'.

- Clicking on the View Details link here shows the overall status of the run, which says success,

The screenshot shows the Google Cloud BigQuery Dataform interface. On the left, there's a sidebar with various options like Analysis, Migration, and Administration. The main area is titled 'Details' and shows a successful workflow execution on Oct 10, 2023, at 10:13:20 PM. It lists four actions: quickstart-source, first\_view, quickstart-table, and second\_view. Each action has a status (Success), start time, duration, and destination. A message at the bottom says 'Successfully created workflow execution'.

- and you can see all of the boilerplate stuff that Dataform added to successfully run this code.

This screenshot shows the same BigQuery Dataform interface, but the 'Actions' section is expanded to show the generated SQL code. The code is a complex BEGIN block with various IF and ELSEIF statements for creating schema, tables, and views based on the source dataset. It includes error handling for existing datasets and permissions.

```

1 BEGIN
2   CREATE SCHEMA IF NOT EXISTS `qwiklabs-gcp-00-1ad27c406e1e.dataform` OPTIONS(location="US");
3   EXCEPTION WHEN ERROR THEN
4     IF NOT CONTAINS_SUBSTR(@error.message, "already exists: dataset") AND
5       NOT CONTAINS_SUBSTR(@error.message, "too many dataset metadata update operations") AND
6       NOT CONTAINS_SUBSTR(@error.message, "User does not have bigquery.datasets.create permission")
7     THEN
8       RAISE USING MESSAGE = @error.message;
9     END IF;
10   END IF;
11   BEGIN
12     | DECLARE dataform_table_type DEFAULT (
13     |   SELECT ANY_VALUE(table_type)
14     |   FROM `qwiklabs-gcp-00-1ad27c406e1e.dataform.INFORMATION_SCHEMA.TABLES`
15     |   WHERE table_name = 'quickstart-table'
16     | );
17     | IF dataform_table_type IS NOT NULL AND dataform_table_type != 'BASE_TABLE' THEN
18     |   IF dataform_table_type = 'BASE_TABLE' THEN
19     |     DROP TABLE IF EXISTS `qwiklabs-gcp-00-1ad27c406e1e.dataform.quickstart-table`;
20     |   ELSEIF dataform_table_type = 'VIEW' THEN
21     |     DROP VIEW IF EXISTS `qwiklabs-gcp-00-1ad27c406e1e.dataform.quickstart-table`;
22     |   ELSEIF dataform_table_type = 'MATERIALIZED_VIEW' THEN
23     |     DROP MATERIALIZED VIEW IF EXISTS `qwiklabs-gcp-00-1ad27c406e1e.dataform.quickstart-table`;
24     |   END IF;
25   END IF;
26   BEGIN
27     CREATE OR REPLACE TABLE `qwiklabs-gcp-00-1ad27c406e1e.dataform.quickstart-table`
  
```

- Clicking on the object here redirects you to the object that was created, and in this case it is the table created with the second SQLX file.

The screenshot shows the Google Cloud BigQuery Workflow Execution page. The workflow has been successfully completed. The details pane shows the start time as Oct 10, 2023, at 10:13:20 PM, status as Success, duration as 8 seconds, source type as Workspace, and source as quickstart-workspace. The actions pane lists four steps: quickstart-source, first\_view, quickstart-table, and second\_view, each with its start time, duration, action, destination, and a 'VIEW DETAILS' link.

- The resulting table can be viewed directly in BigQuery, under the **Preview** tab.

The screenshot shows the Google Cloud BigQuery Explorer interface. The left sidebar displays workspace resources, including a dataform named 'quickstart-table'. The main panel shows the 'PREVIEW' tab for the 'quickstart-table'. The table has columns 'fruit' and 'count', with data rows: 1 apples (count 2), 2 pears (count 1), 3 bananas (count 0), and 4 oranges (count 5).

| Row | fruit   | count |
|-----|---------|-------|
| 1   | apples  | 2     |
| 2   | pears   | 1     |
| 3   | bananas | 0     |
| 4   | oranges | 5     |