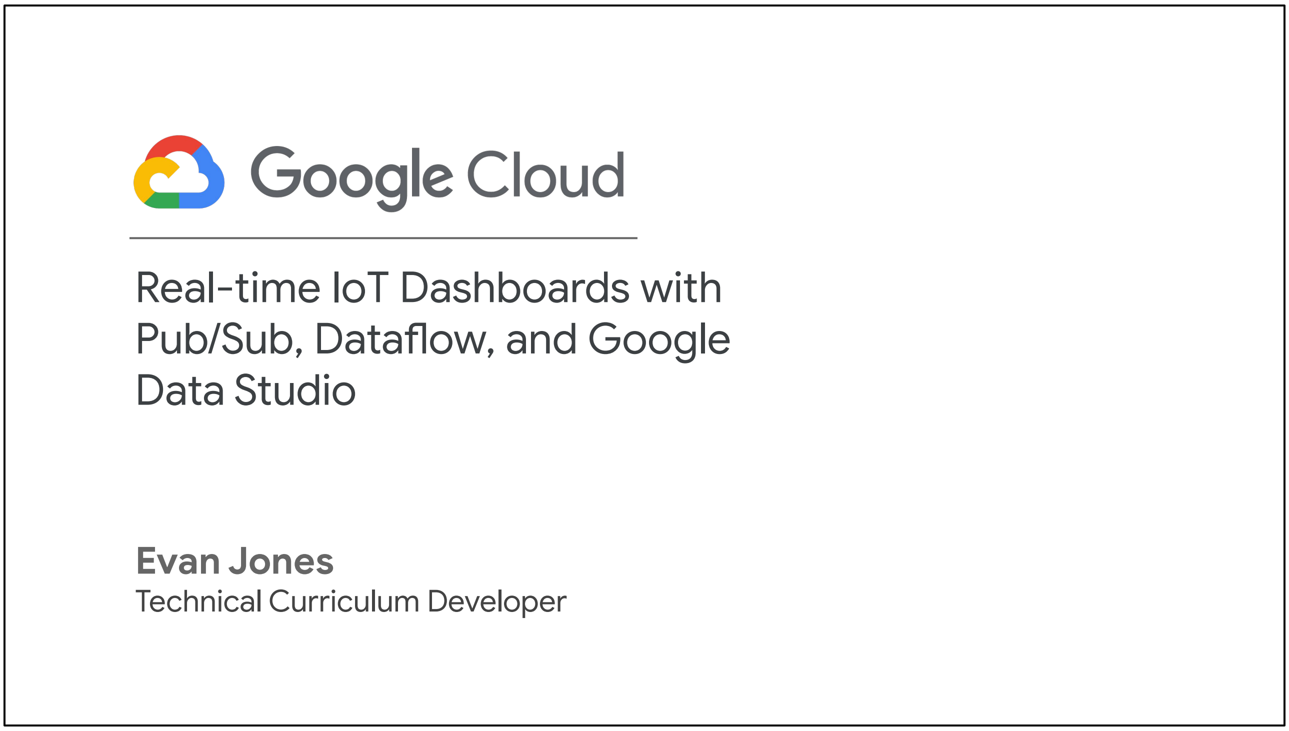


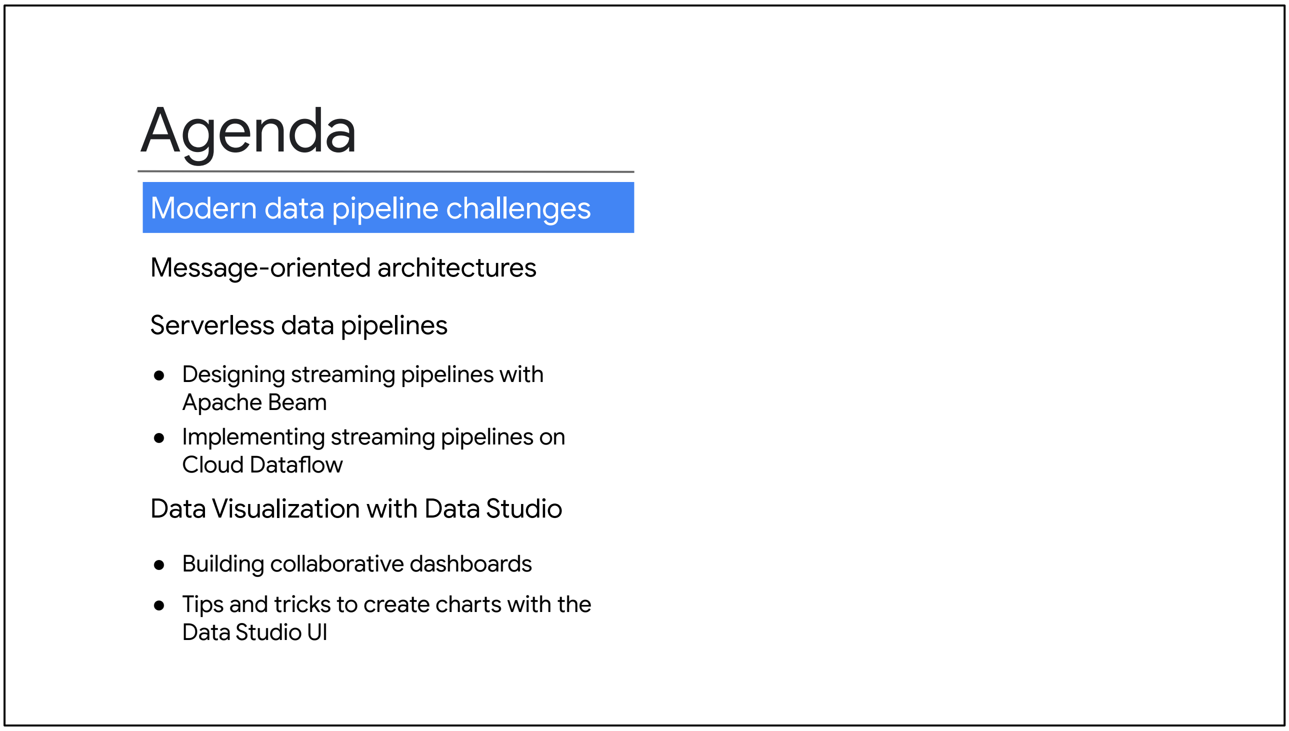
Google Cloud

Real-time IoT Dashboards with Pub/Sub, Dataflow, and Google Data Studio

Evan Jones
Technical Curriculum Developer

So far in the course our data has been loaded in batch form from pre-existing structured databases like Cloud SQL or BigQuery. But what about if the data is on thousands of sensors world-wide? And what if they send new data messages every minute? How can you harness such upstream data sources for your analysis?

Welcome to the module on Real-time IoT Dashboards where we will highlight and solve the challenges of streaming data processing.



Agenda

Modern data pipeline challenges

Message-oriented architectures

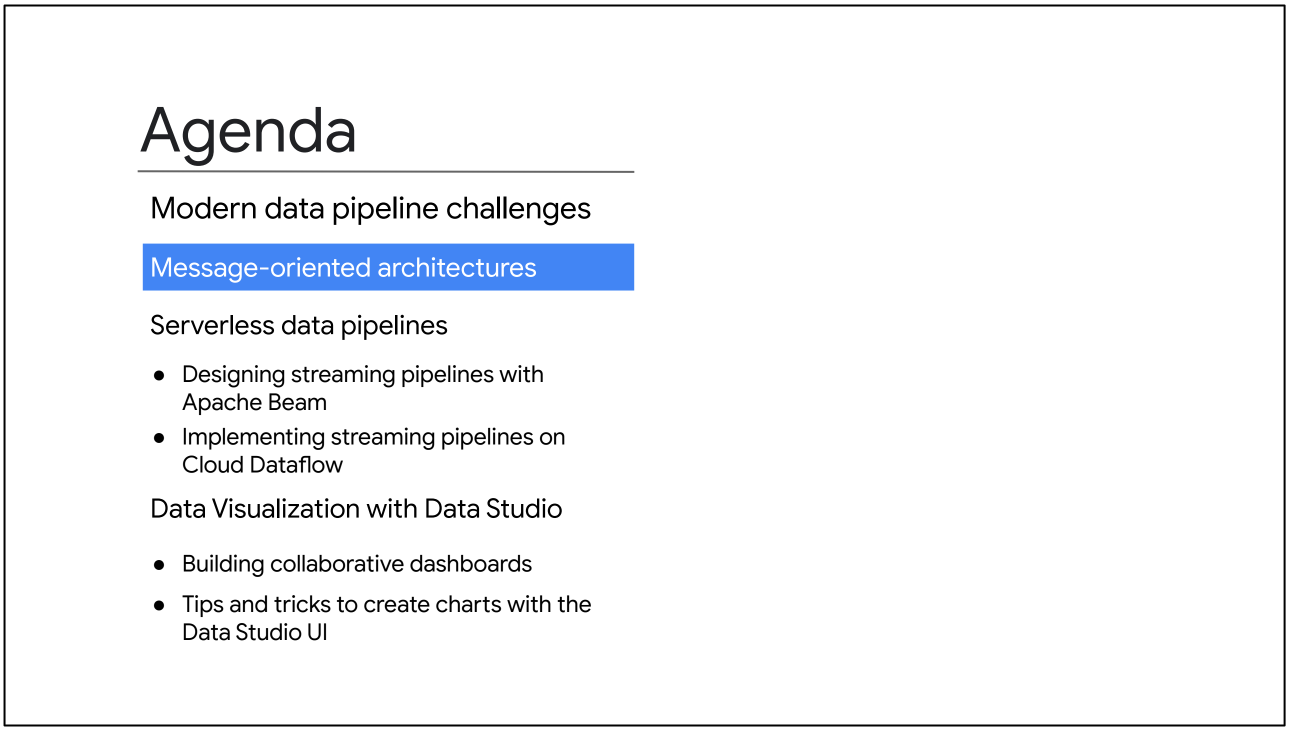
Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

We will first look at the challenges that give today's data engineers headaches when trying to setup and manage their pipelines



Agenda

Modern data pipeline challenges

Message-oriented architectures

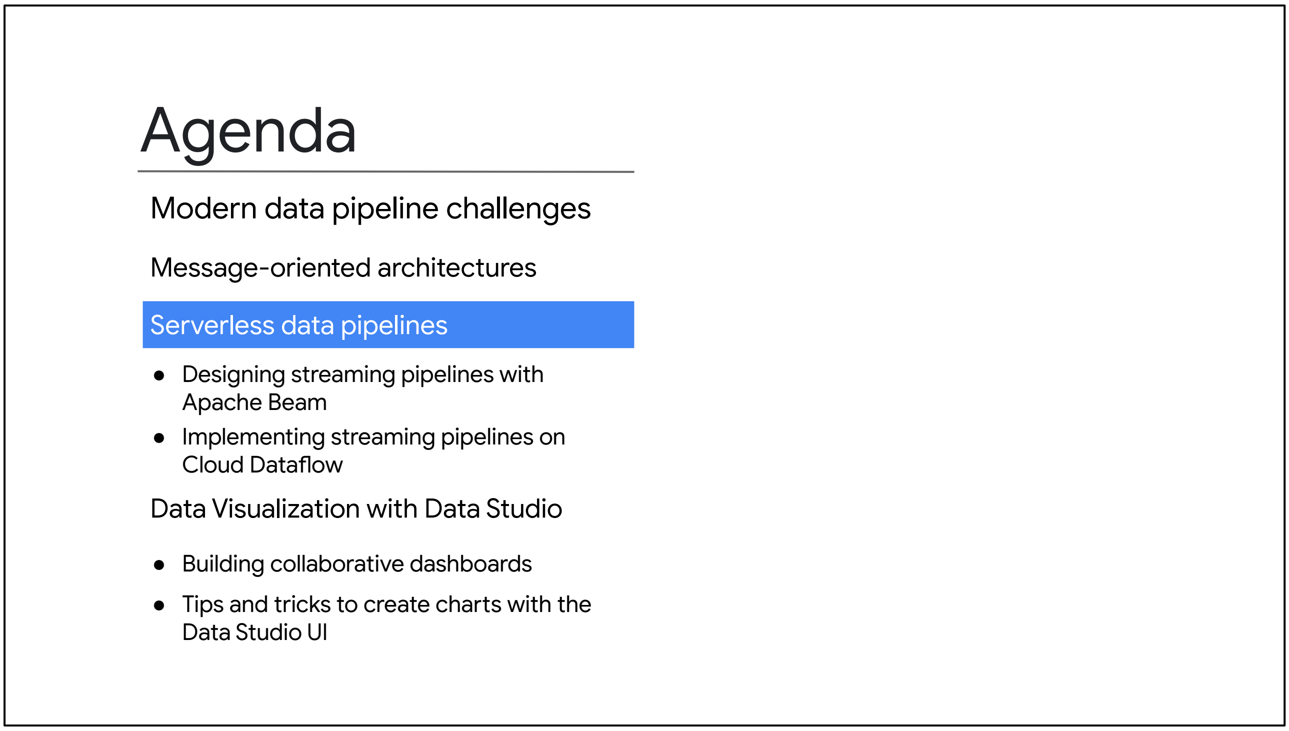
Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

We will then examine how we can capture all of these streaming messages and wrangle them globally, reliably, at scale, and then feed them into our pipeline.



Agenda

Modern data pipeline challenges

Message-oriented architectures

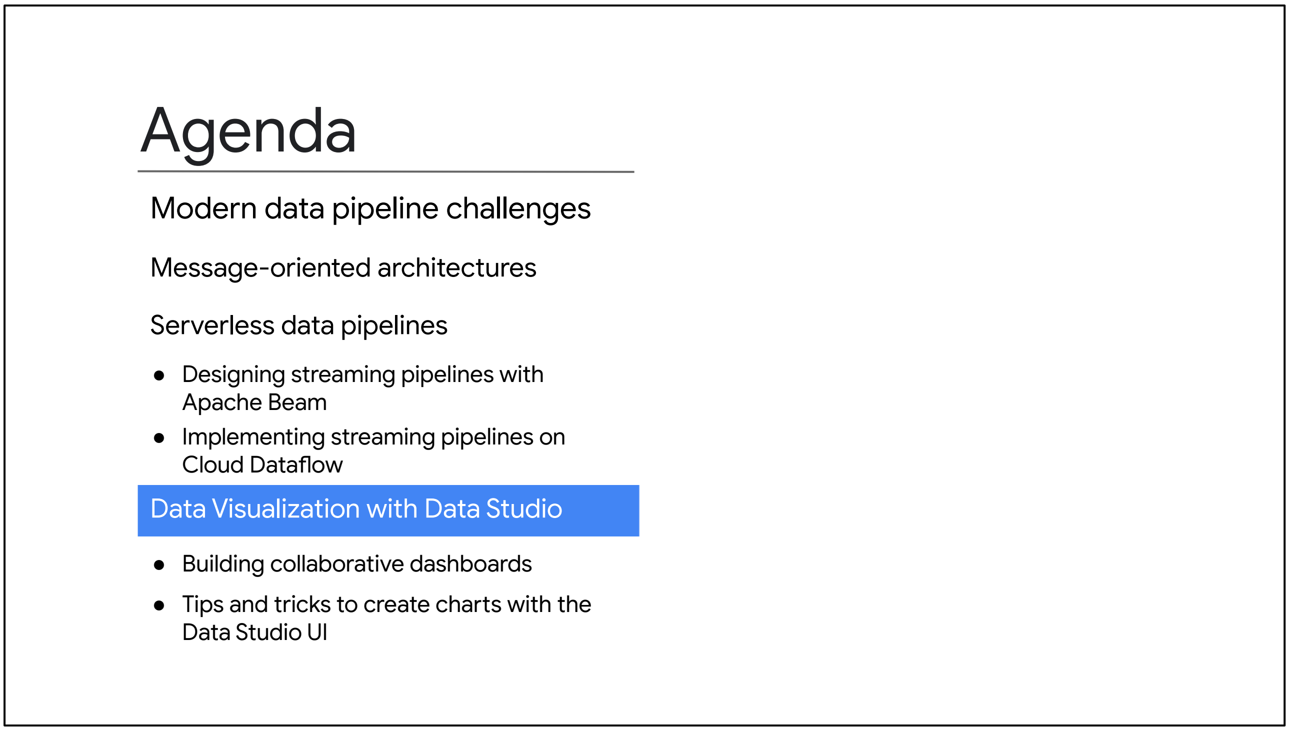
Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

After we’ve captured the streaming messages, we will show you how you can build serverless data pipelines for streaming data using Google Cloud Platform. You may be asking, serverless? How can I design and control the pipeline for scale if I need to? We’ll cover all of that in our sections on Apache Beam and Cloud Dataflow.



Agenda

Modern data pipeline challenges

Message-oriented architectures

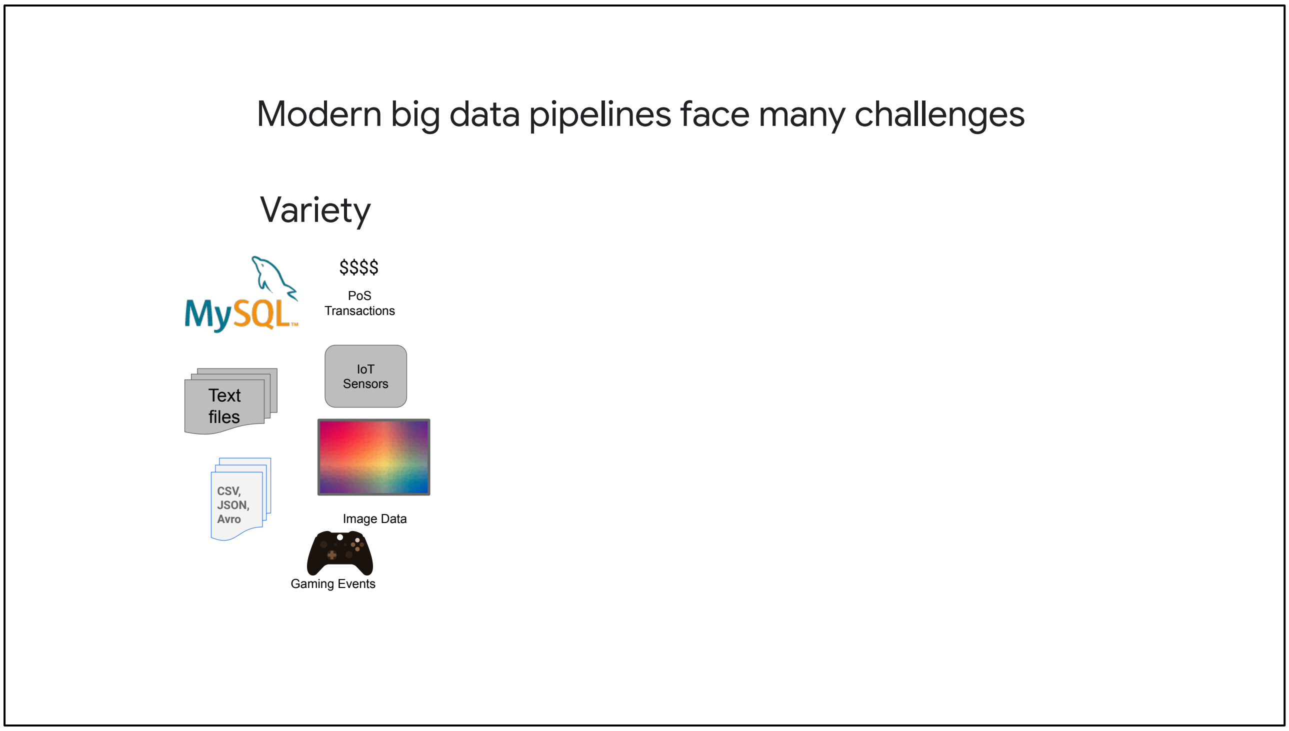
Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

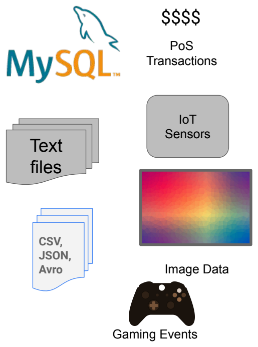
- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

Lastly, once we’ve captured, processed, and stored the data -- we’ll address how we can visualize our insights into reporting dashboards. [pause]



Modern big data pipelines face many challenges

Variety

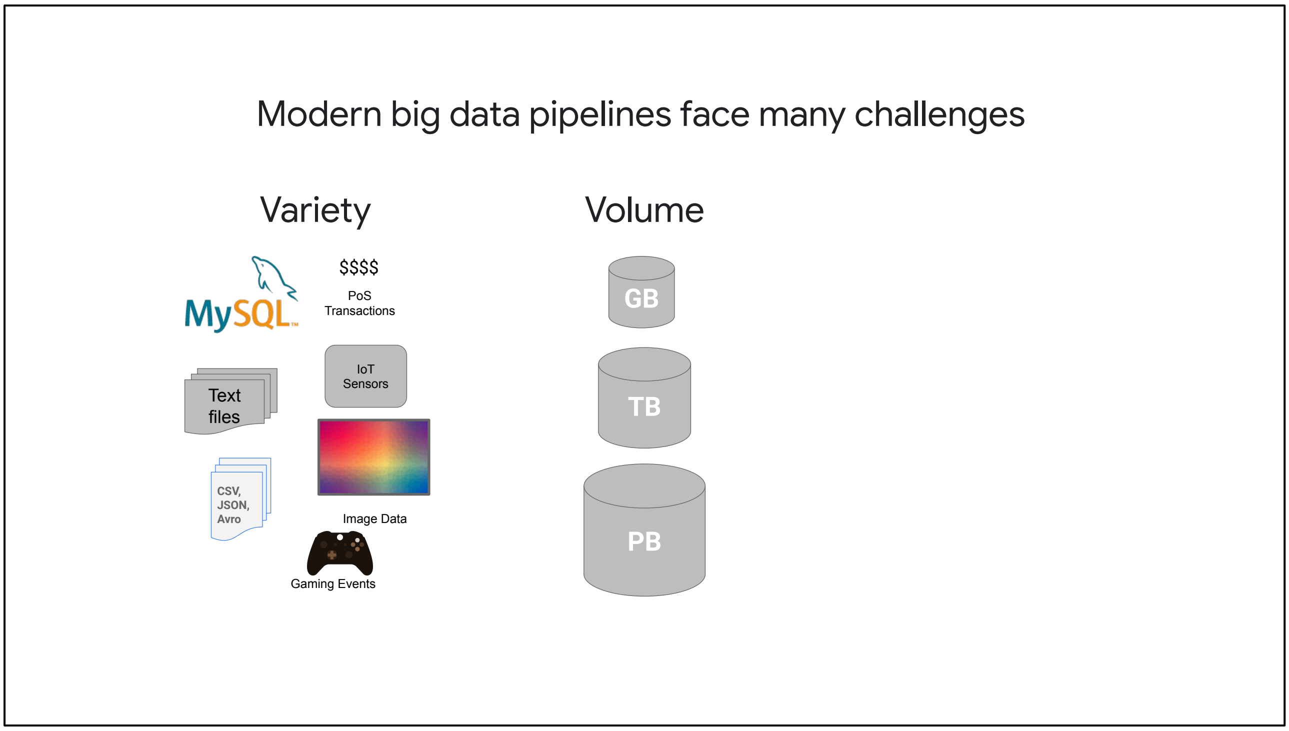


Building scalable and reliable pipelines is a core responsibility of data engineers. In modern organizations, data could come in from all different sources.

Imagine a gaming company that tracks user's high scores as events and needs to log progress toward player objectives.

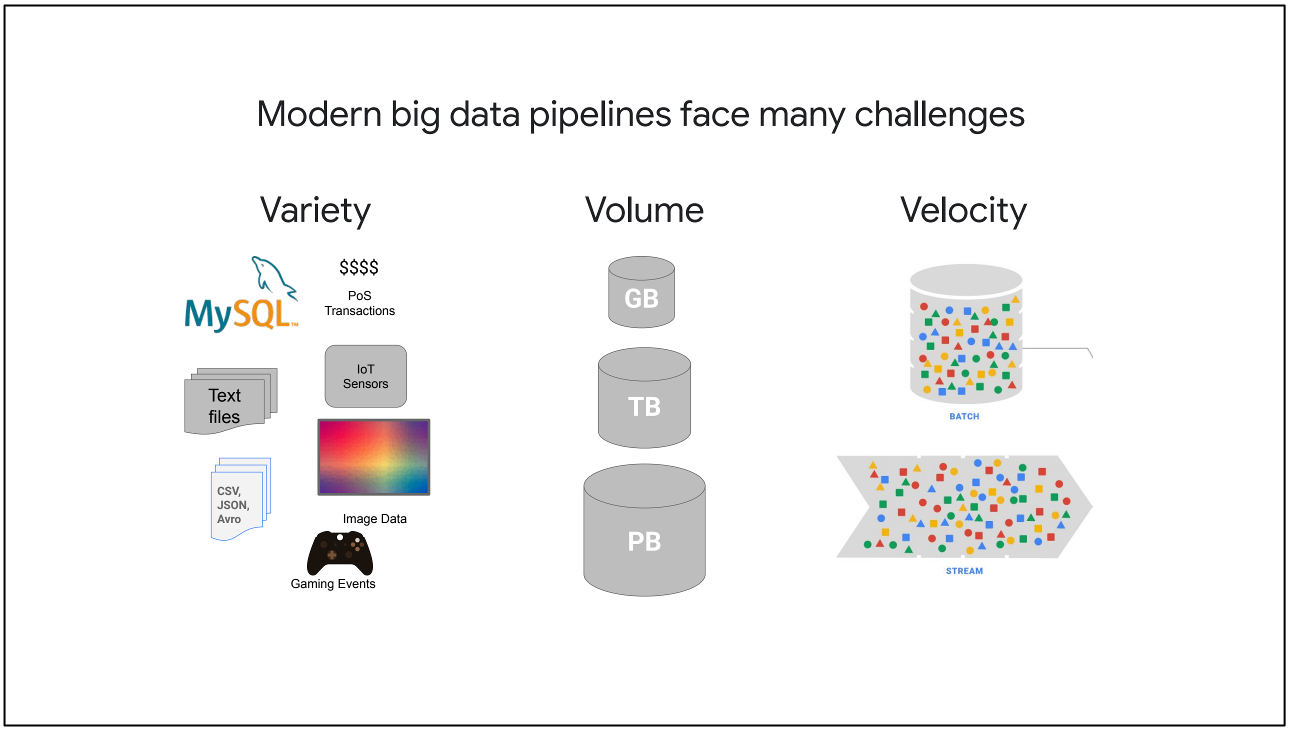
Or hundreds of distributed IoT sensors reporting weather readings from around the globe.

What about point of sale data from a thousand storefronts? How do we alert our downstream systems of new orders in an organized way with no duplicates?



Modern big data pipelines face many challenges

Now let's amplify the magnitude of the challenge to handle not only an arbitrary variety of input sources but a variable volume of data that could start as gigabytes and scale up to petabytes. Will our pipeline code and infrastructure scale with it or will the pipeline grind to a halt or crash?

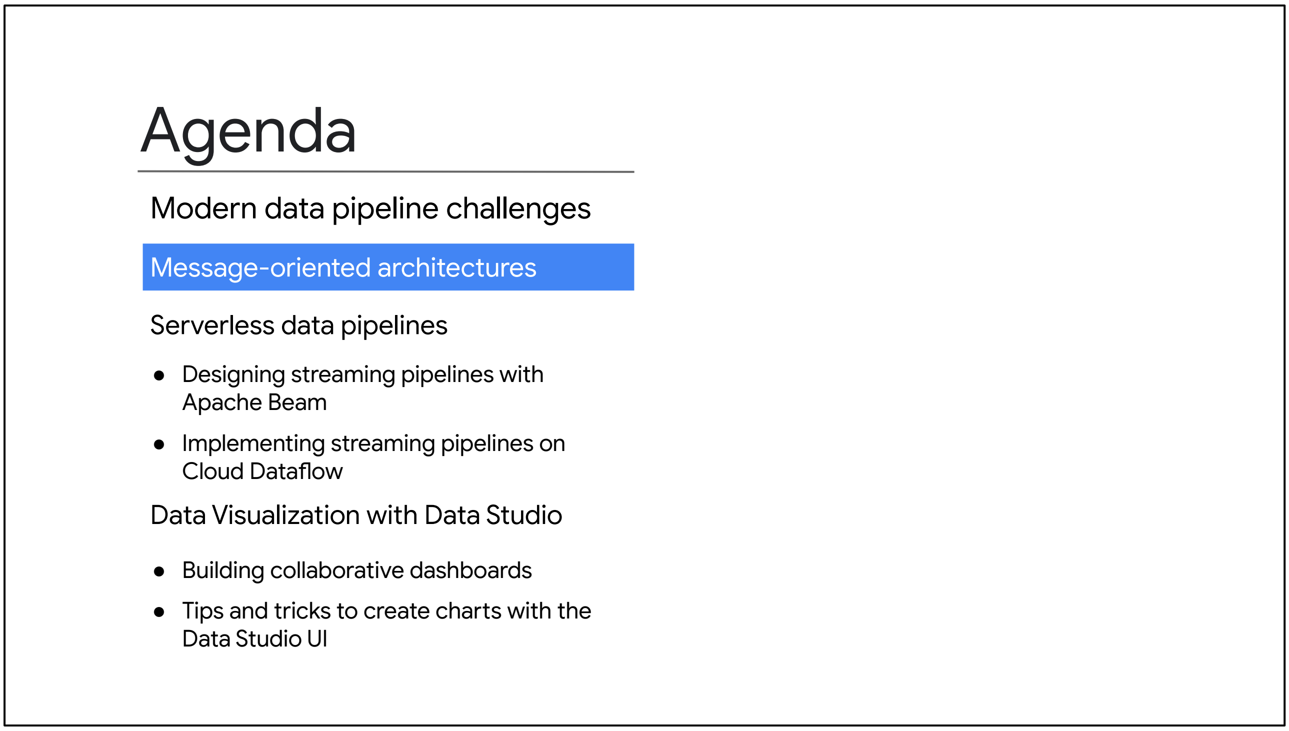


Modern big data pipelines face many challenges

And we need to do it all in near real-time as soon as the data touches our system.

Oh and we need a way to handle if the data arrives late, has bad data in the message, or needs to be transformed mid-flight before we stream it into our warehouse.

As you can start to see there are significant challenges at every turn for the pipeline developer. We'll unpack each of these challenges and hopefully, by the end of this module, you'll not only have the peace of mind knowing that there are solutions out there but also the real world experience building a streaming data pipeline in your lab.



Agenda

Modern data pipeline challenges

Message-oriented architectures

Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

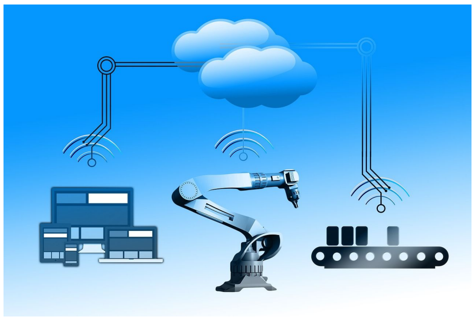
Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

Let’s now discuss one of the first pieces in the pipeline puzzle which is handling large volumes of streaming data that may not be coming from a single structured database. Instead these events messages could be streaming in from a thousand or a million different events all happening asynchronously.

IoT devices present new challenges to data ingestion

Distributed Messages

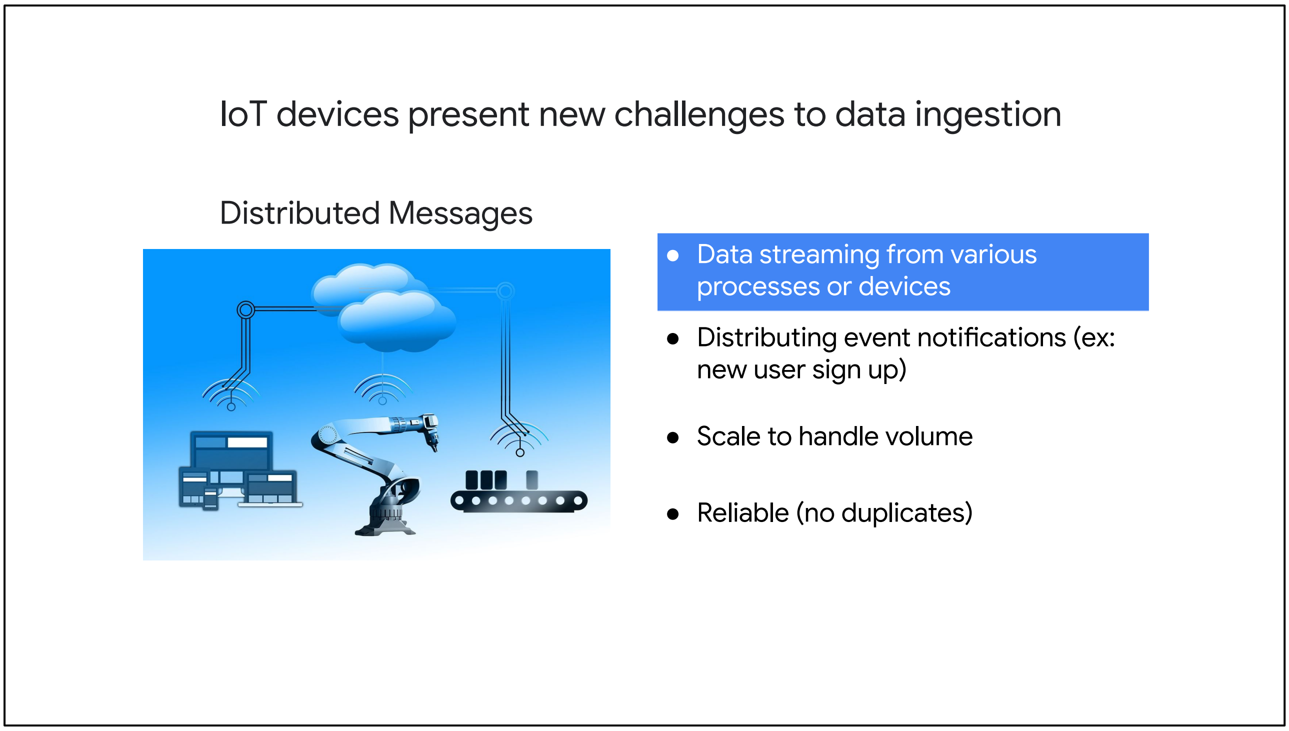


- Data streaming from various processes or devices
- Distributing event notifications (ex: new user sign up)
- Scale to handle volume
- Reliable (no duplicates)

A common use case where you see this pattern occur is with IoT or Internet of Things applications. IoT devices could be sensors on the Go-JEK drivers' motorcycles that you saw earlier sending out location data every 30 seconds for every driver. Or, they could be temperature sensors planted throughout a datacenter to help optimize heating and cooling.

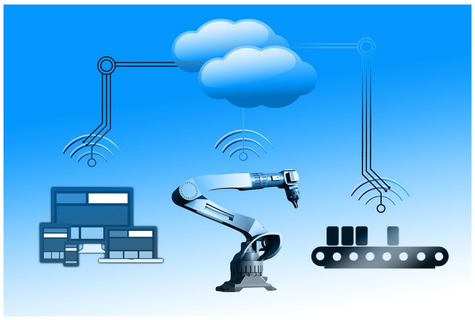
Whatever the use case we have to tackle these four components.

https://cloud.google.com/pubsub/docs/overview#scenarios



IoT devices present new challenges to data ingestion

Distributed Messages

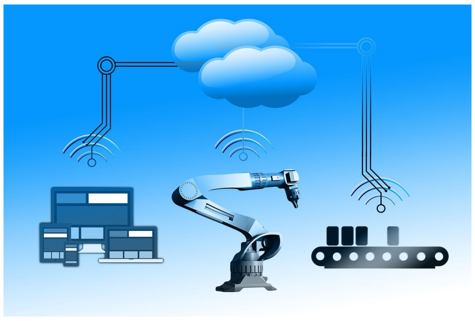


- Data streaming from various processes or devices
- Distributing event notifications (ex: new user sign up)
- Scale to handle volume
- Reliable (no duplicates)

Data is streaming from various processes or devices that may not talk to each other and even may be sending in bad data or data that is delayed.

IoT devices present new challenges to data ingestion

Distributed Messages

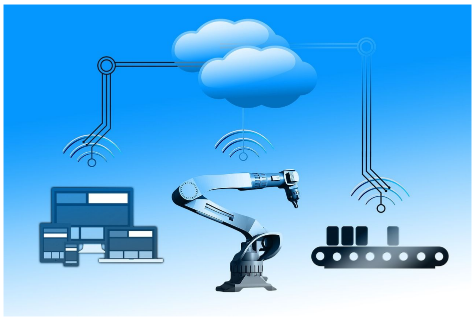


- Data streaming from various processes or devices
- Distributing event notifications (ex: new user sign up)
- Scale to handle volume
- Reliable (no duplicates)

We need to have a way not only to collect these streaming messages but also allow other services to subscribe to new messages that we publish.

IoT devices present new challenges to data ingestion

Distributed Messages

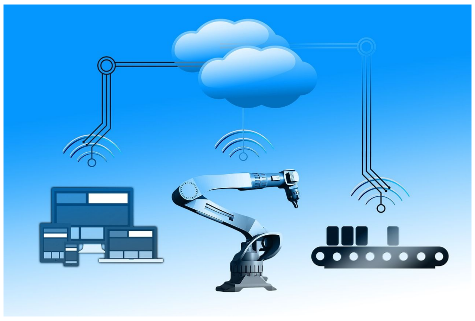


- Data streaming from various processes or devices
- Distributing event notifications (ex: new user sign up)
- Scale to handle volume
- Reliable (no duplicates)

Naturally, the service needs to handle an arbitrarily high amount of data so we don't lose any messages coming in

IoT devices present new challenges to data ingestion

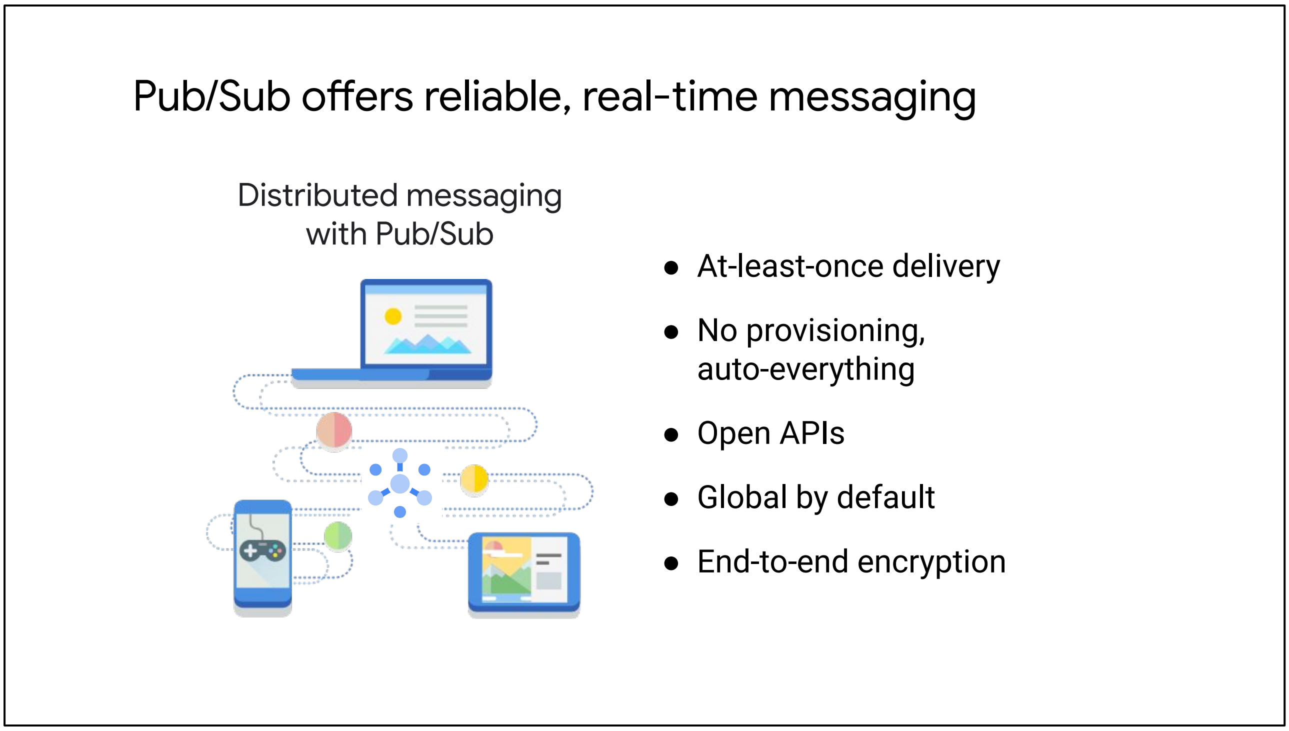
Distributed Messages



- Data streaming from various processes or devices
- Distributing event notifications (ex: new user sign up)
- Scale to handle volume
- Reliable (no duplicates)

And it has to be reliable. We need all the messages collected and also a way to remove any duplicates if found.

https://cloud.google.com/pubsub/docs/overview#scenarios



Pub/Sub offers reliable, real-time messaging

Distributed messaging with Pub/Sub

- At-least-once delivery
- No provisioning, auto-everything
- Open APIs
- Global by default
- End-to-end encryption

One tool to handle distributed message-oriented architectures at scale is Cloud Pub/Sub. The name is easy to remember because it’s Publisher / Subscriber or publish messages to subscribers.

Pub/sub is a distributed messaging service that can receive messages from a variety of device streams like gaming events, IoT devices, application streams and more.

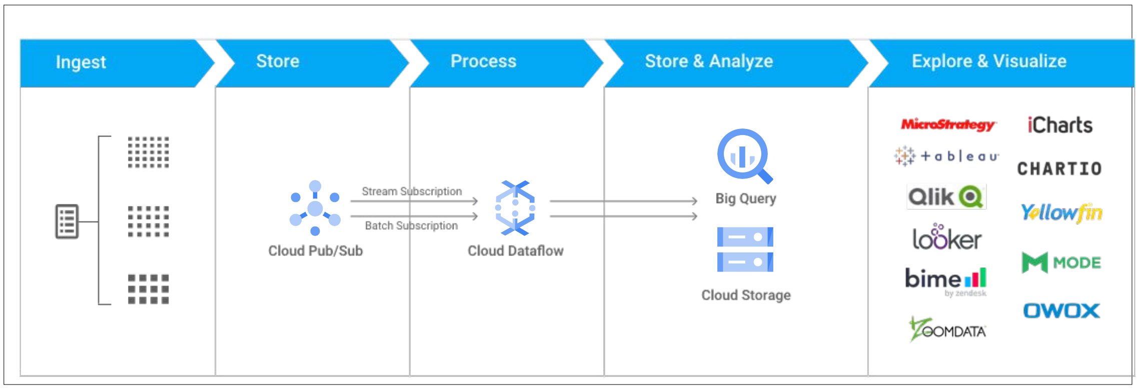
It ensures at-least once delivery of messages received to subscribing applications and no provisioning is required -- whether its a ton of messages or none pub/sub will scale to meet it

The APIs are open and the service is global by default and offers end-to-end encryption.

https://cloud.google.com/pubsub/docs/overview#scenarios

https://cloud.google.com/pubsub/

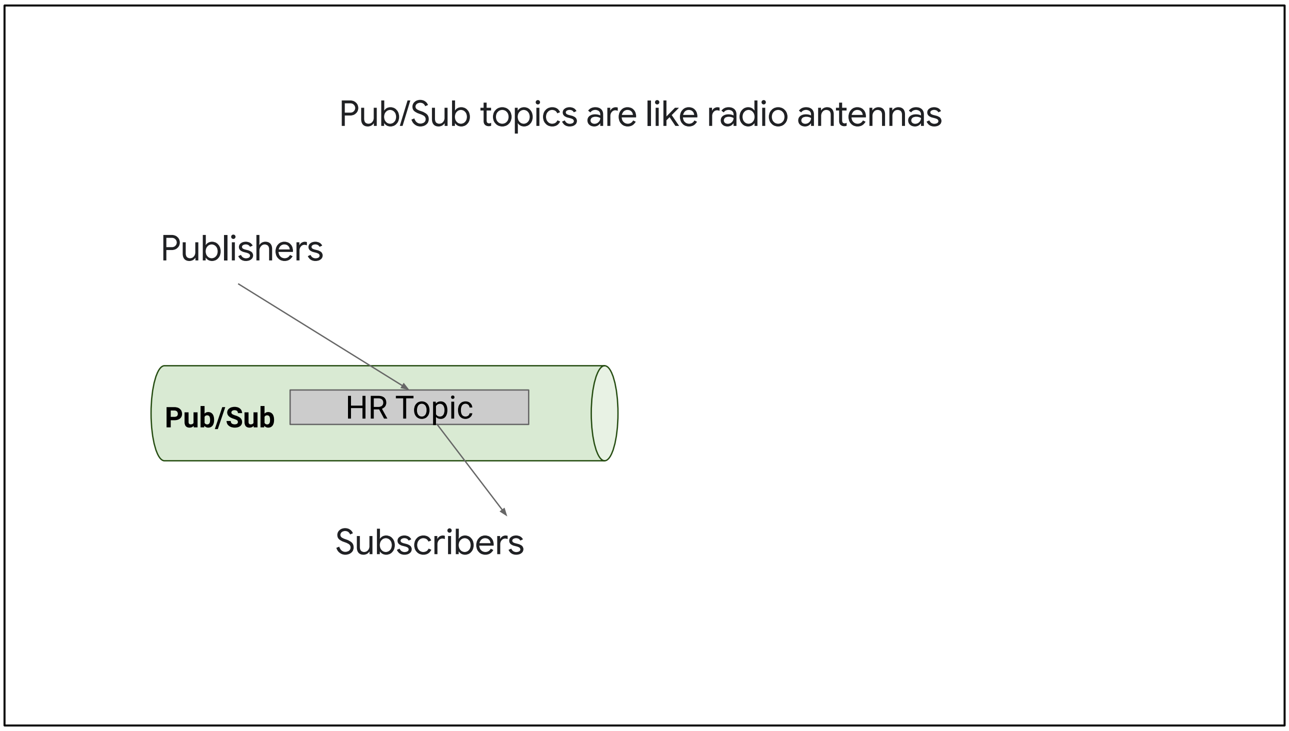
Google Cloud's serverless big-data pipeline



Here's what the end-to-end architecture looks like. Upstream source data starts in from the left from devices all over the globe and is ingested into Cloud Pub/Sub which is the first point of contact within our system. Cloud Pub/Sub reads, stores, and publishes out to any subscribers of this data topic that new messages are available.

Cloud Dataflow, as a subscriber of pub/sub, can ingest and transform those messages in an elastic streaming pipeline and output the results into an analytics data warehouse like BigQuery. Lastly, you can connect a data visualization tool like Tableau, Looker, or Data Studio to visualize and monitor the results of the pipeline.

Next we'll talk more about the architecture of Pub/Sub.



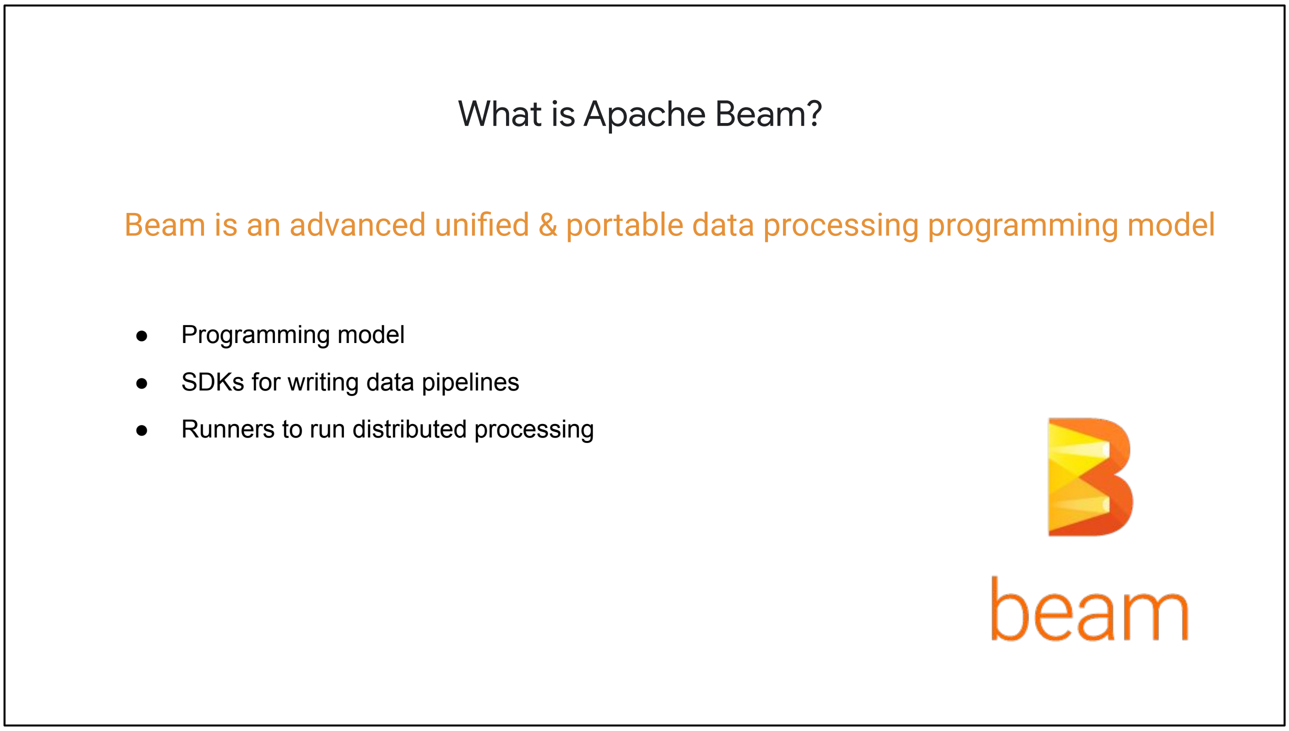
A central piece of pub/sub is the Topic. You can think of a topic like a radio antenna. Whether your radio is blasting music or turned off the antenna itself is always there.

If music is being broadcasted on a frequency that nobody is listening to -- the stream of music still exists. Similarly, a Publisher can send data to a topic that has no Subscriber to receive it.

Or a subscriber can be waiting for data from a topic that isn't getting data sent to it. Like listening to all that static from a bad radio frequency.

Or could could have a fully operational pipeline where the Publisher is sending data to a topic that an application is subscribed to.

To recap: There can be zero, one, or more publishers, and zero, one or more subscribers related to a topic. And they're completely decoupled, so they're free to break without affecting their counterparts. Let's look at an example.



What is Apache Beam?

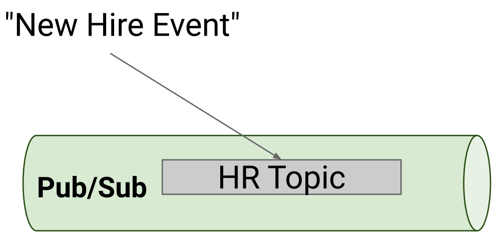
Beam is an advanced unified & portable data processing programming model

- Programming model
- SDKs for writing data pipelines
- Runners to run distributed processing

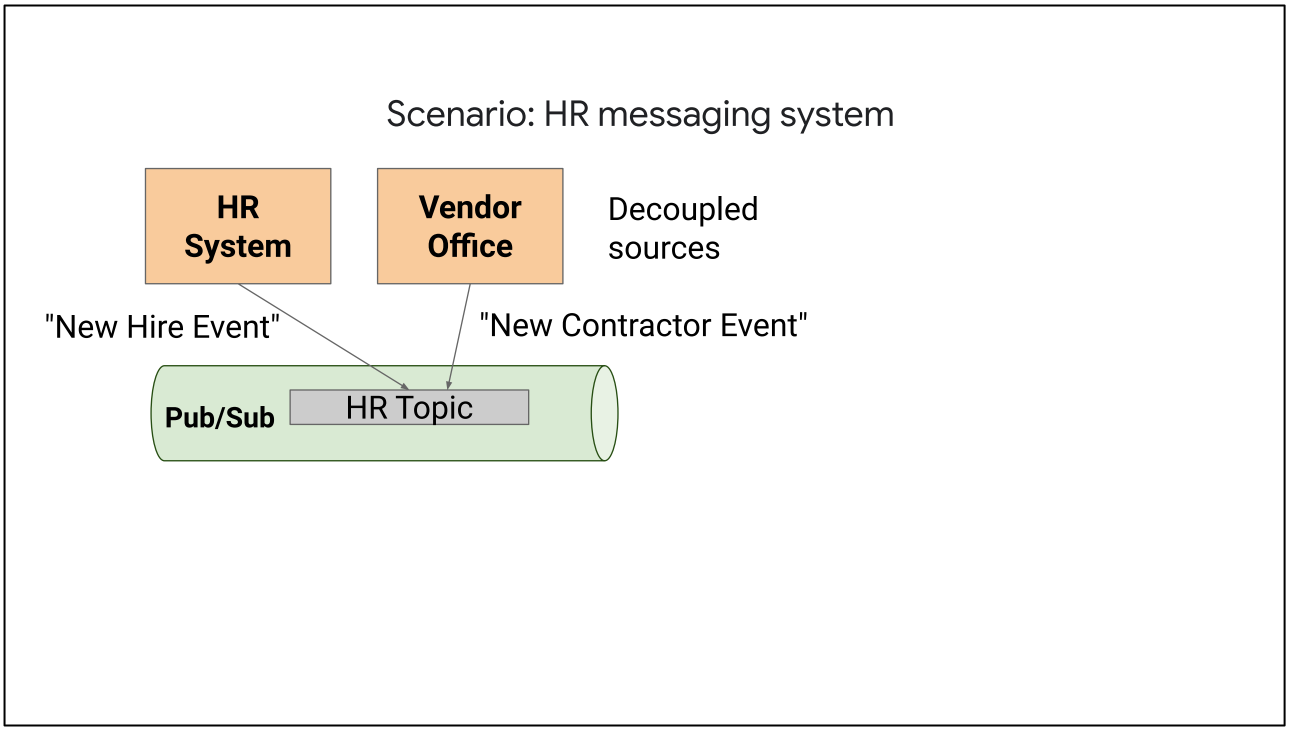
beam

Apache beam is a portable data processing programming model. It’s open source and can be ran in a highly distributed fashion.

Scenario: HR messaging system

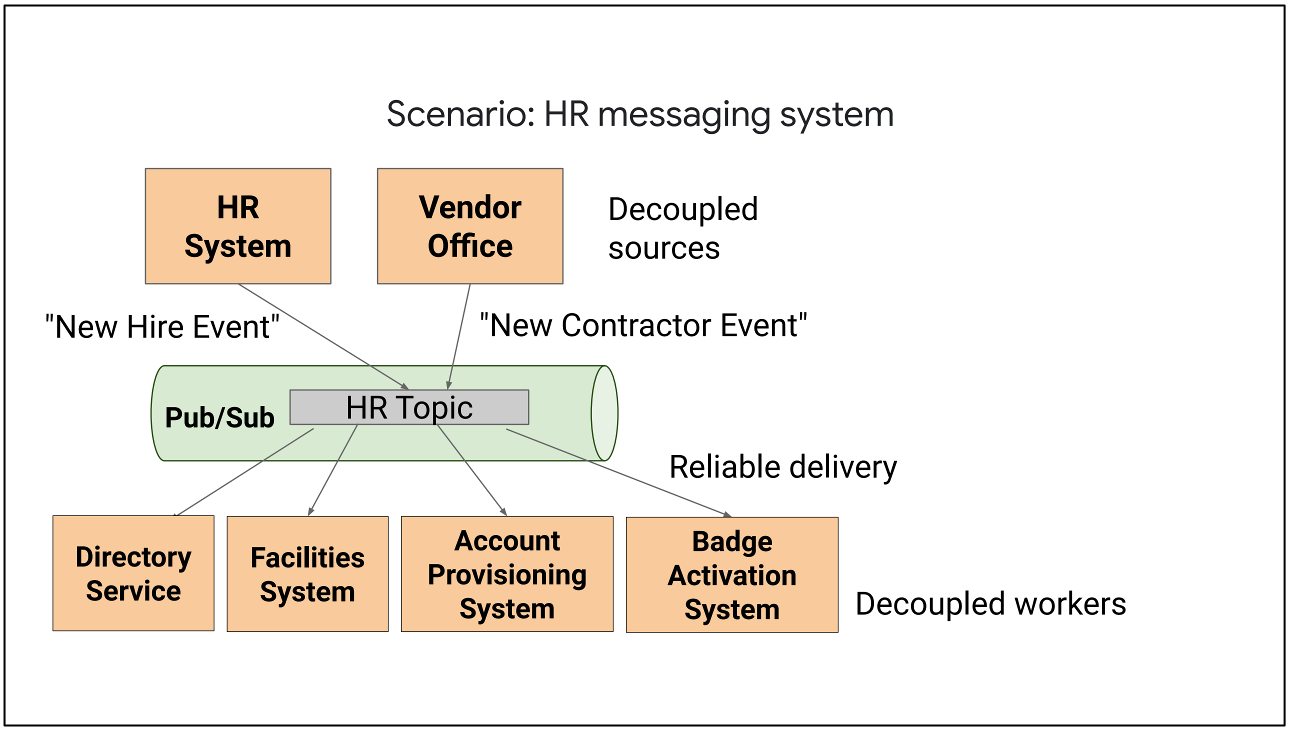


Say you’ve got an HR topic. A new person joins your company and this notification should allow other applications that care about a new user joining to subscribe and get that message. What applications could tell you that a new person joined?



Well you have two different types of workers: full time employee or contractor. Both sources could have no-knowledge of the other but still publish their events saying “this person joined” into the Pub/Sub HR Topic.

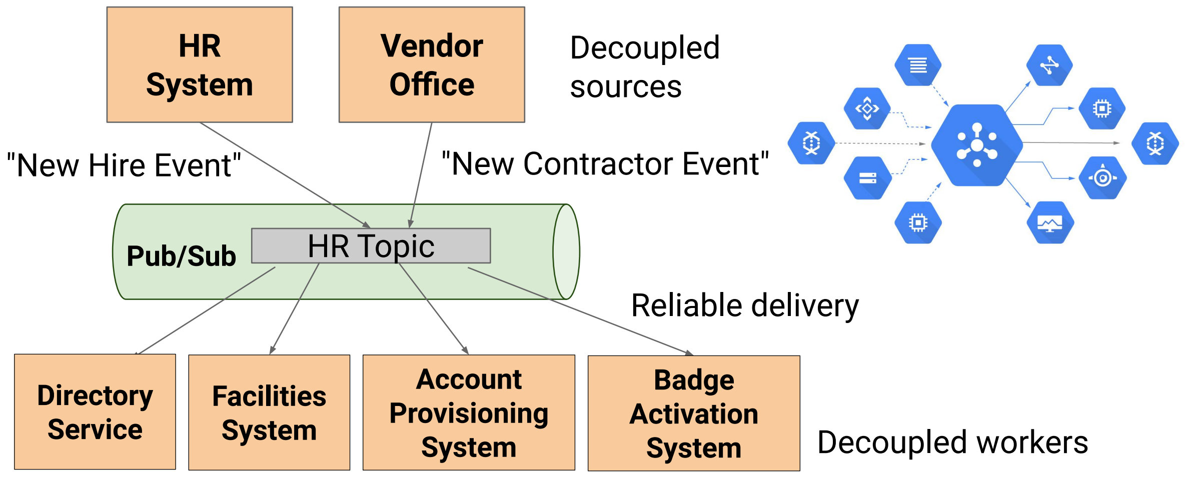
What other areas of your business would like to know as soon as a new person joins?



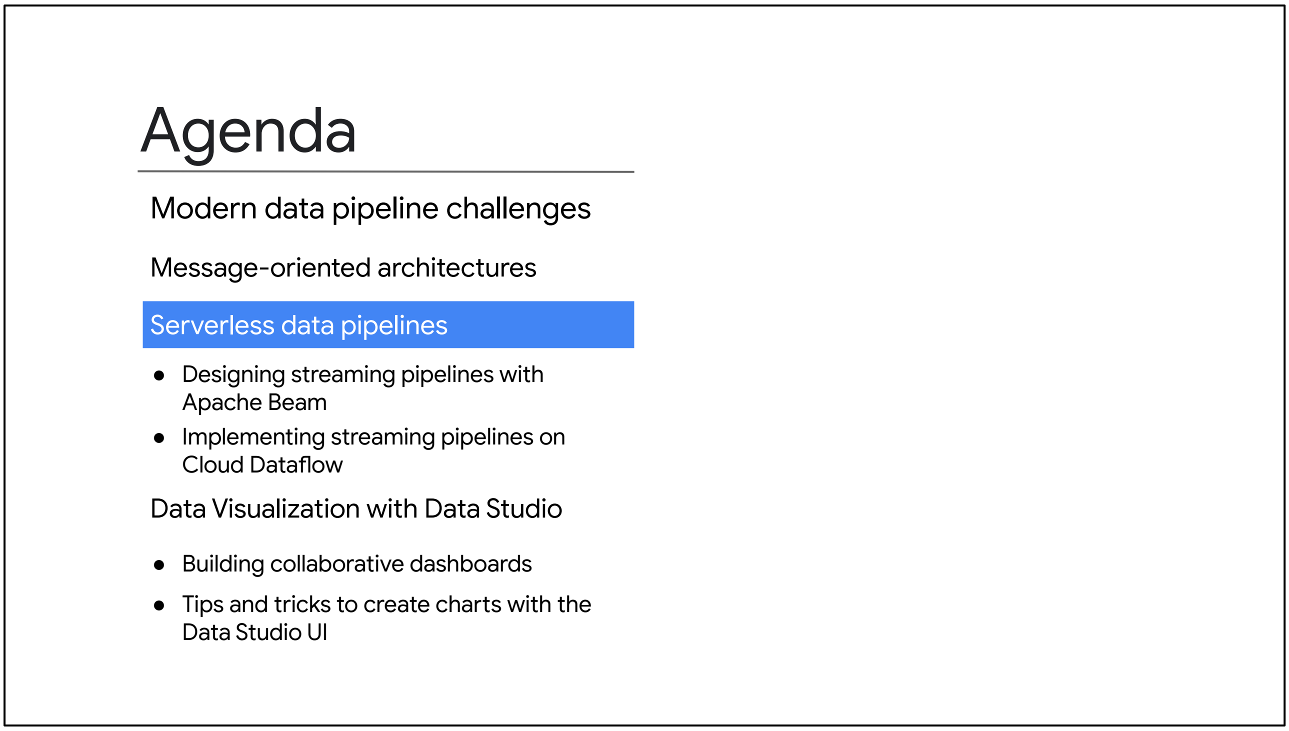
Once Pub/Sub receives the message, downstream applications like the directory service, facilities system, account provisioning, and badge activation systems can all listen and process their own next steps independent of one another.

Pub/Sub is a good solution to buffer changes for lightly-coupled architectures like this one here with many different sources and sinks.

Scenario: HR messaging system



It supports many different inputs and outputs and you can even publish one pub/sub event from a topic to yet a different pub sub topic should you wish. For our application we want to get these messages reliably into our data warehouse and we're going to need a pipeline that can match Pub/Sub's scale and elasticity to do it.



Agenda

Modern data pipeline challenges

Message-oriented architectures

Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

Now that we’ve captured all of our messages from a variety of streaming input sources we need a way to pipe that data into our data warehouse for analysis.

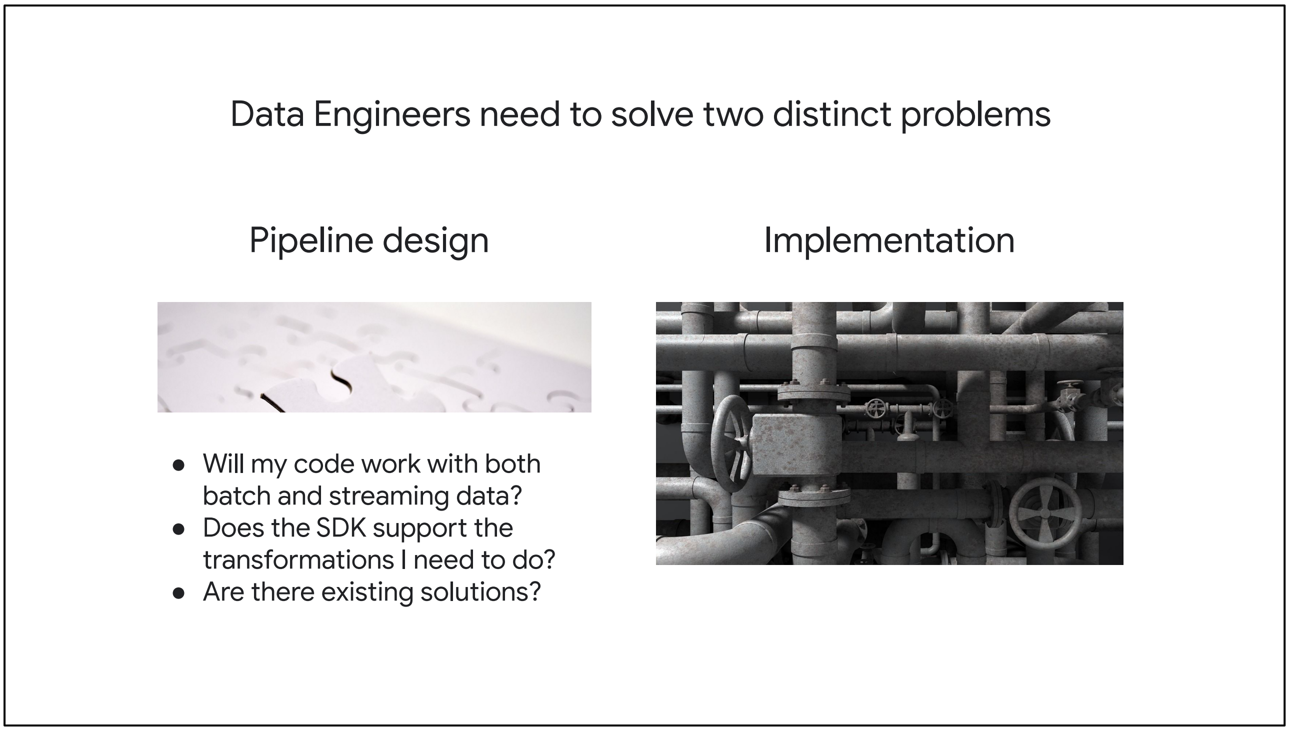
Data Engineers need to solve two distinct problems

Pipeline design

Implementation

Here’s a quick mental framework for the two problems we will need to solve as a data engineer building our pipeline. Much like an architect drawing a blueprint and then eventually building a house, we have to solve for:

1) The design of the pipeline in code
2) The actual implementation and serving of the pipeline at scale



Data Engineers need to solve two distinct problems

Pipeline design

- Will my code work with both batch and streaming data?
- Does the SDK support the transformations I need to do?
- Are there existing solutions?

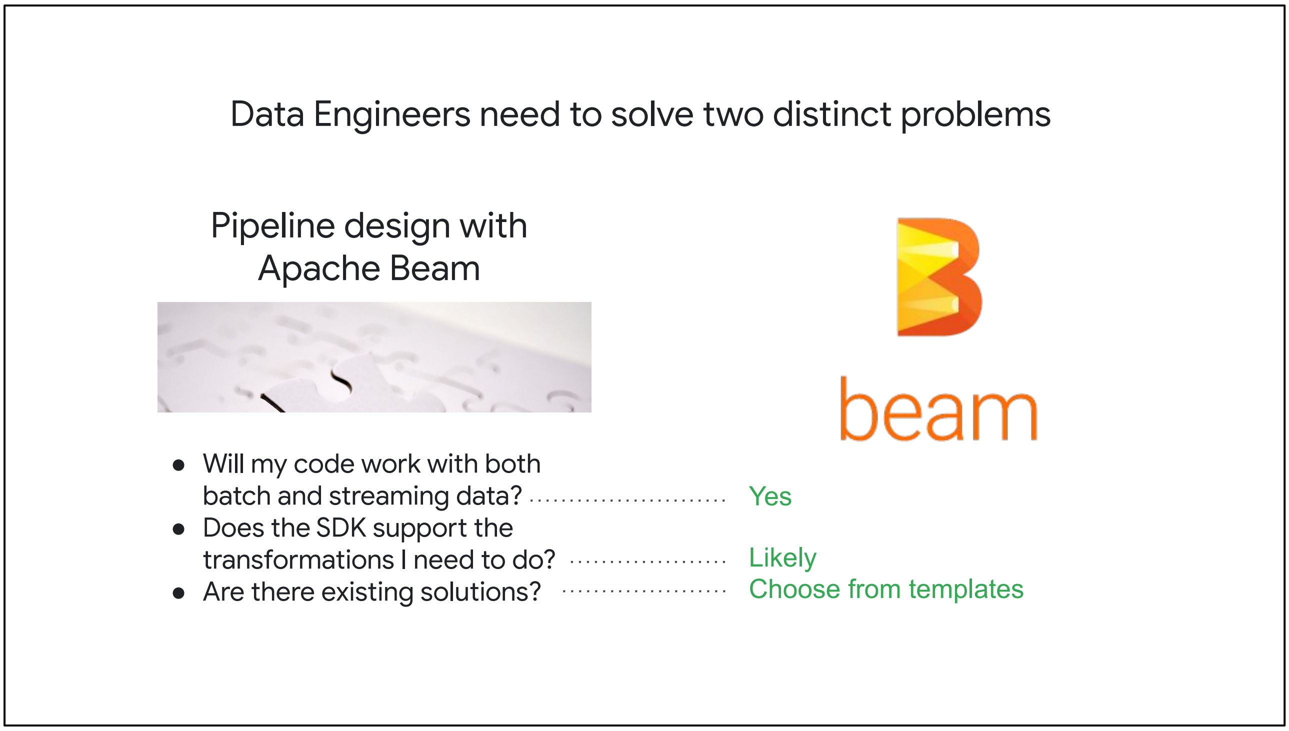
Implementation

Here are some common questions in each of these phases

For design, is the code we're going to write compatible with both batch and streaming data or do we need to do any refactoring?

Does the pipeline code SDK we're using have all the transformations, mid-flight aggregations and windowing, and the ability to handle late data?

Lastly, for design, are there existing templates and solutions we could reference to quickly get started?



Data Engineers need to solve two distinct problems

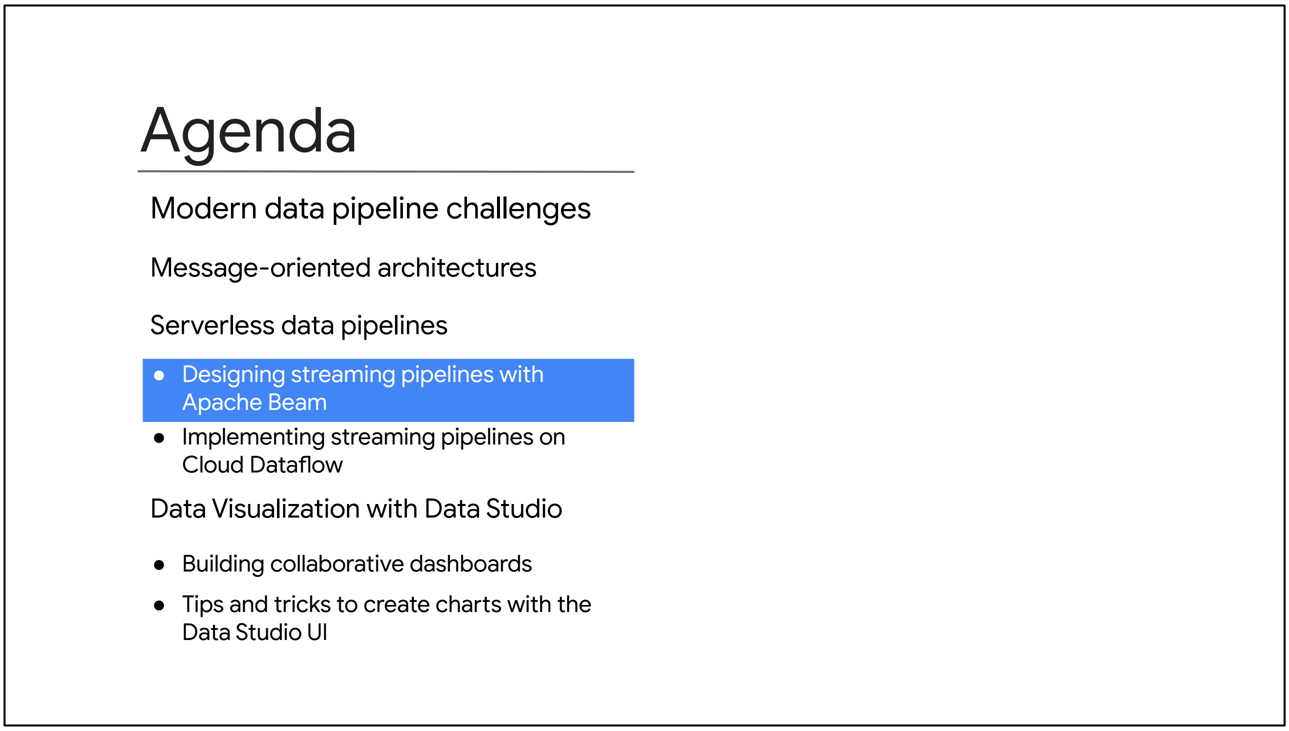
Pipeline design with Apache Beam

● Will my code work with both batch and streaming data?	Yes
● Does the SDK support the transformations I need to do?	Likely
● Are there existing solutions?	Choose from templates

beam

One popular open source solution that we talked about earlier in the course is Apache Beam.

It works with both batch and streaming data and has an ever-expanding list of transformations and functions committed to it's open source repository. Best of all for this class is you will be starting from existing templates for your pipeline code.



Agenda

Modern data pipeline challenges

Message-oriented architectures

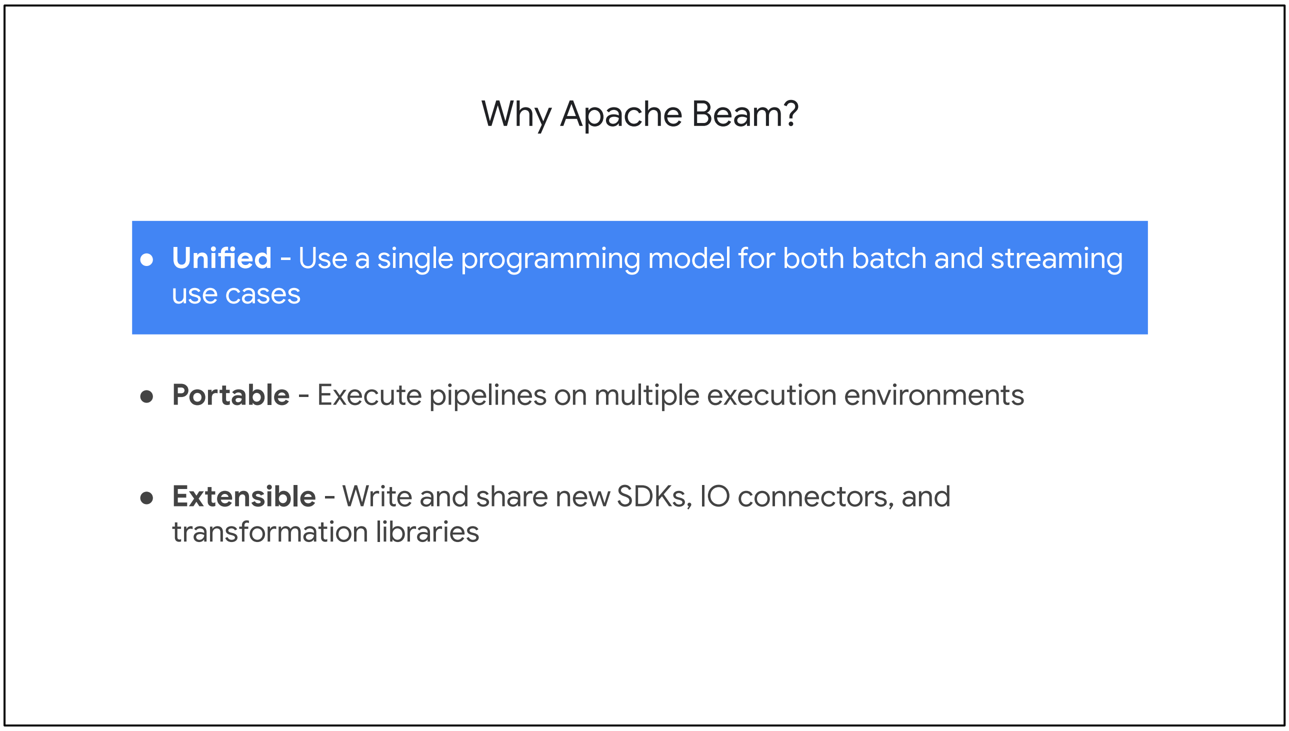
Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

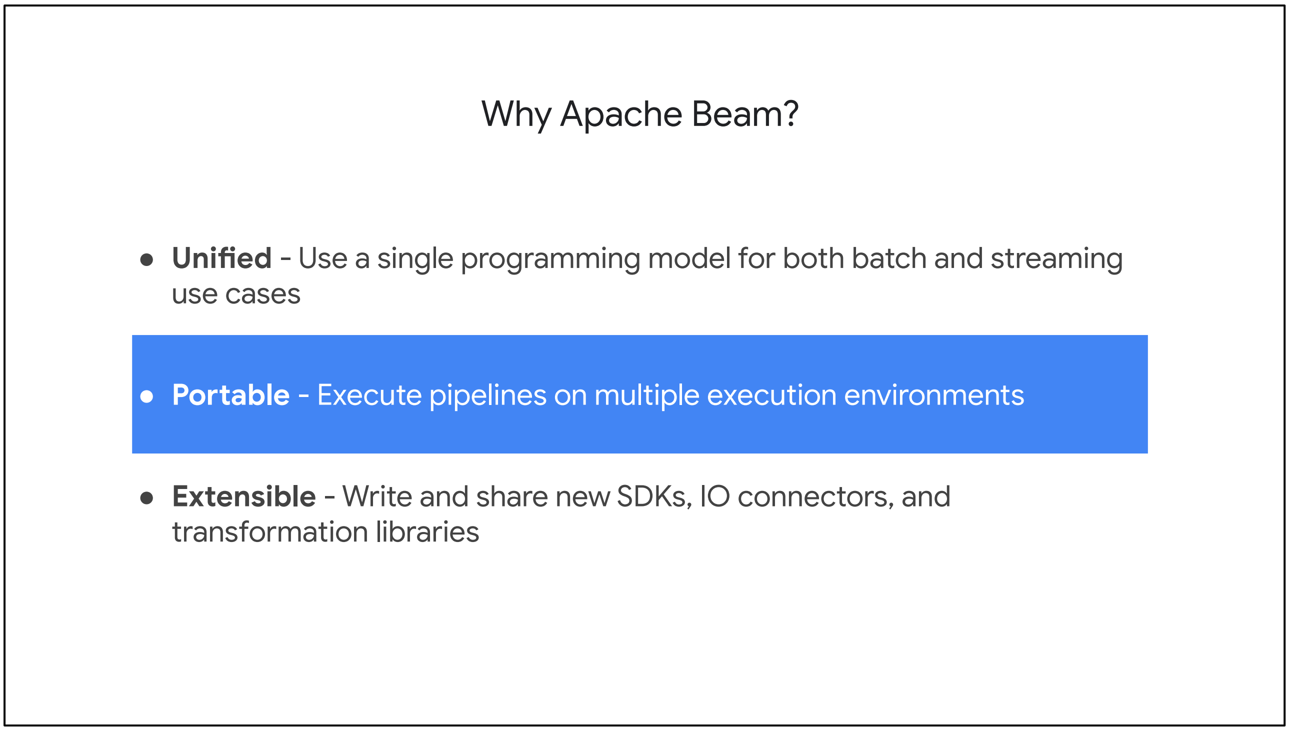
So what exactly is Apache beam and what does a pipeline written using the SDK look like?



Why Apache Beam?

- **Unified** - Use a single programming model for both batch and streaming use cases
- **Portable** - Execute pipelines on multiple execution environments
- **Extensible** - Write and share new SDKs, IO connectors, and transformation libraries

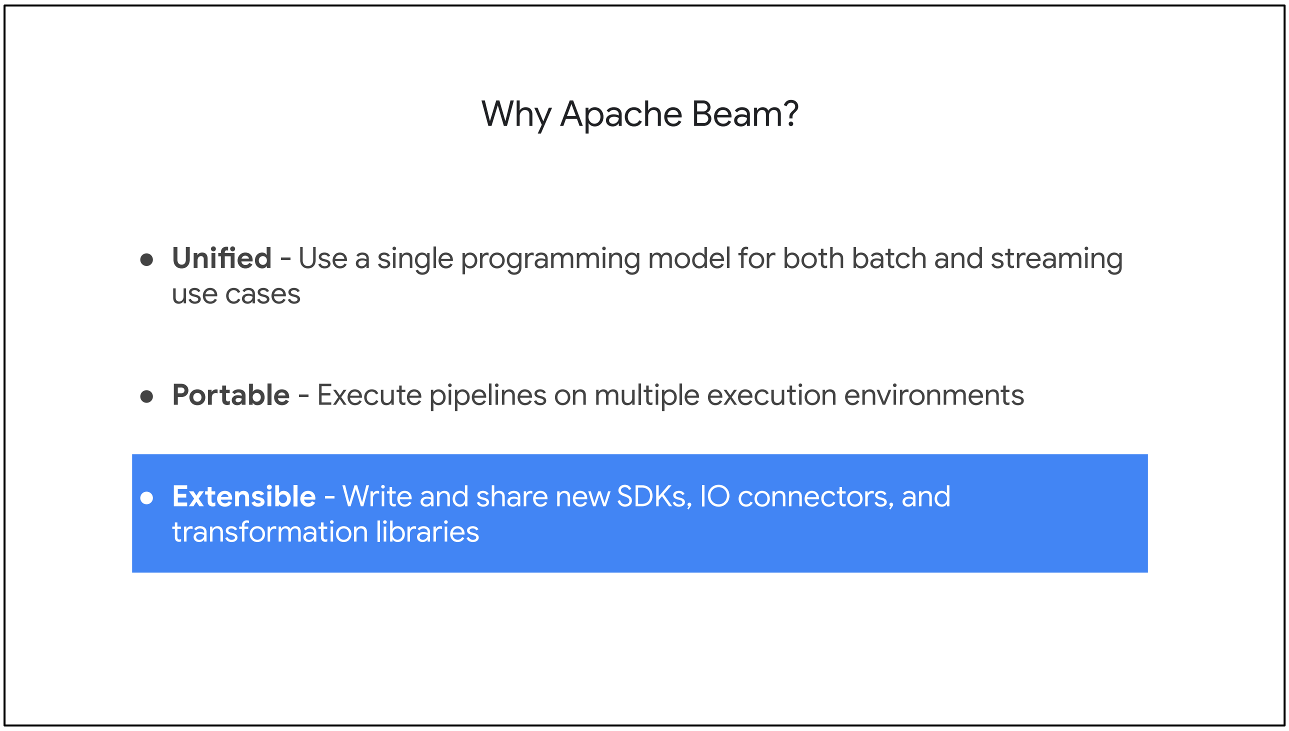
It’s unified in that a single model (meaning your pipeline code) can work for both batch and streaming data



Why Apache Beam?

- **Unified** - Use a single programming model for both batch and streaming use cases
- **Portable** - Execute pipelines on multiple execution environments
- **Extensible** - Write and share new SDKs, IO connectors, and transformation libraries

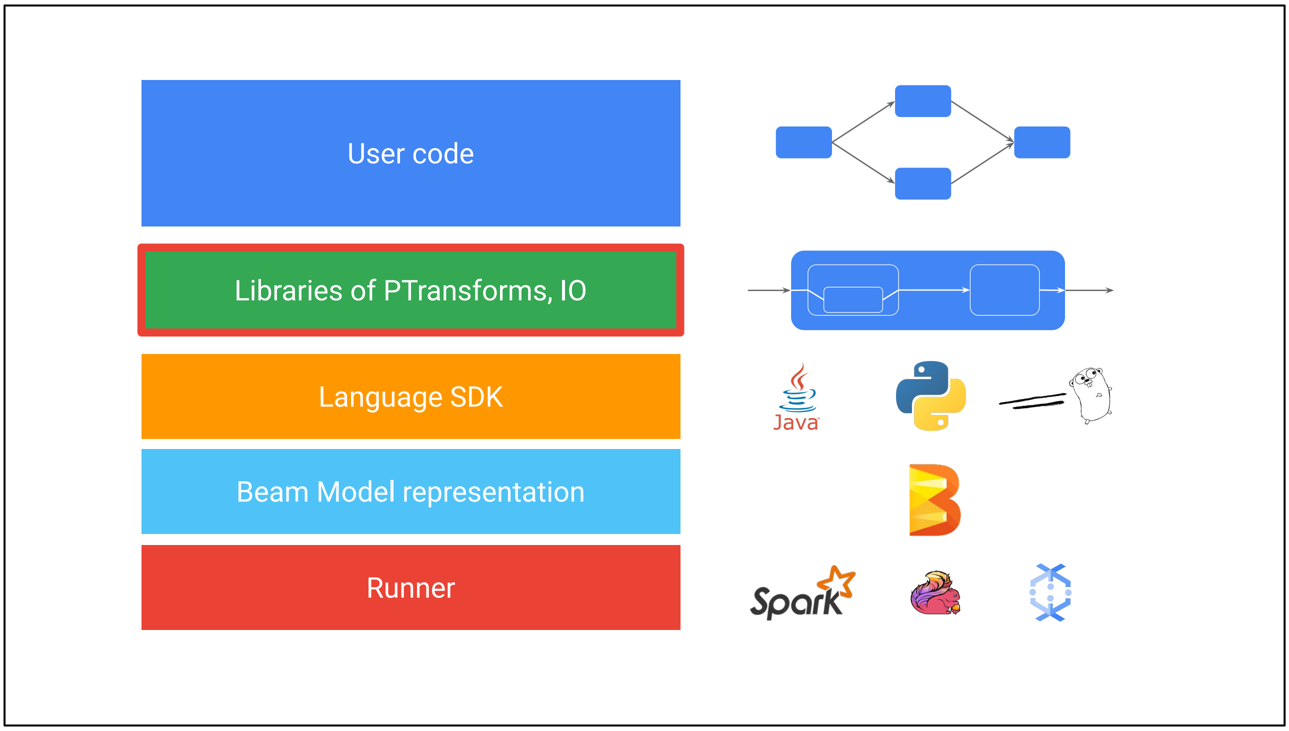
You're not locked into a vendor because the code is portable and can work on multiple execution environments like Cloud Dataflow, Apache Spark and others.



Why Apache Beam?

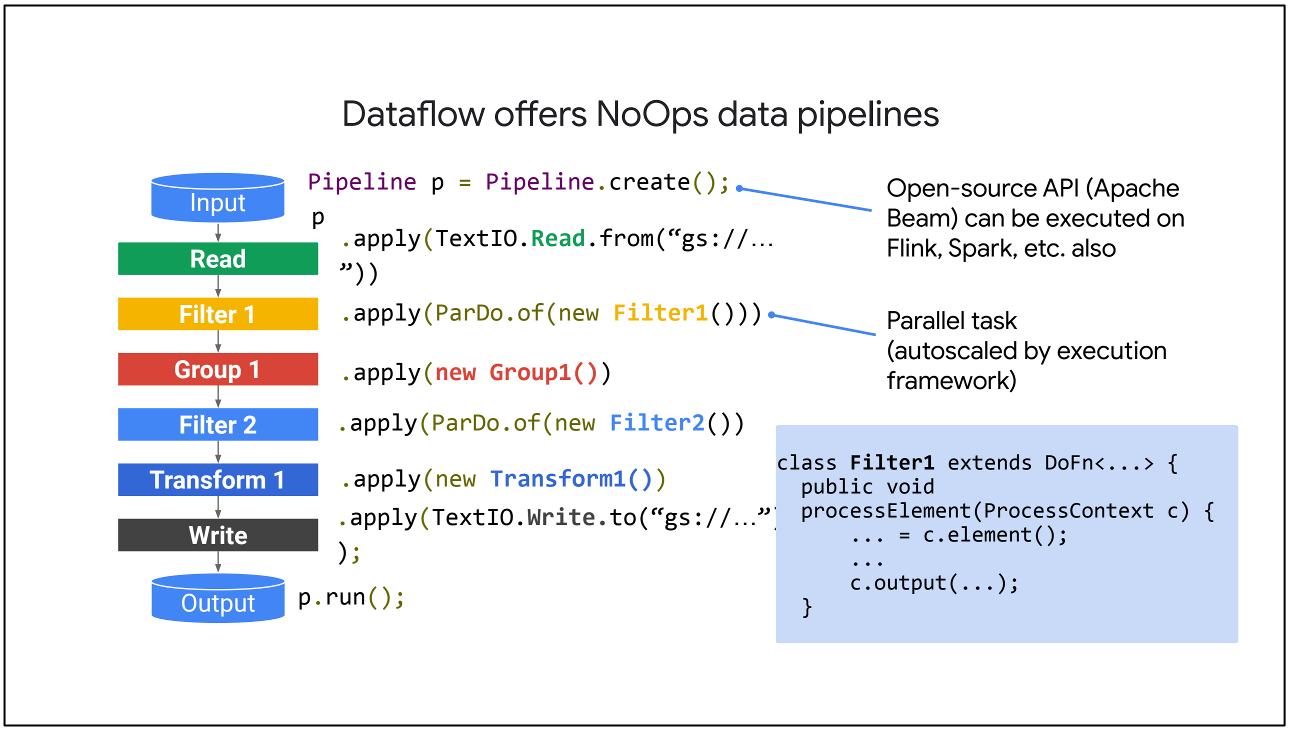
- **Unified** - Use a single programming model for both batch and streaming use cases
- **Portable** - Execute pipelines on multiple execution environments
- **Extensible** - Write and share new SDKs, IO connectors, and transformation libraries

It’s extensible and open source. You can browse and write your own connectors and transformation libraries if you wanted.



Apache Beam pipelines are written in Java, Python or Go. The SDK provides a host of libraries for transformations and data connectors to sources and sinks.

Apache Beam creates a model representation from your code which is portable across many runners. Runners pass off your model for execution on a variety of different possible engines. Cloud Dataflow being a popular choice.



Dataflow offers NoOps data pipelines

Input
Read
Filter 1
Group 1
Filter 2
Transform 1
Write
Output
Open-source API (Apache Beam) can be executed on Flink, Spark, etc. also
Parallel task (autoscaled by execution framework)

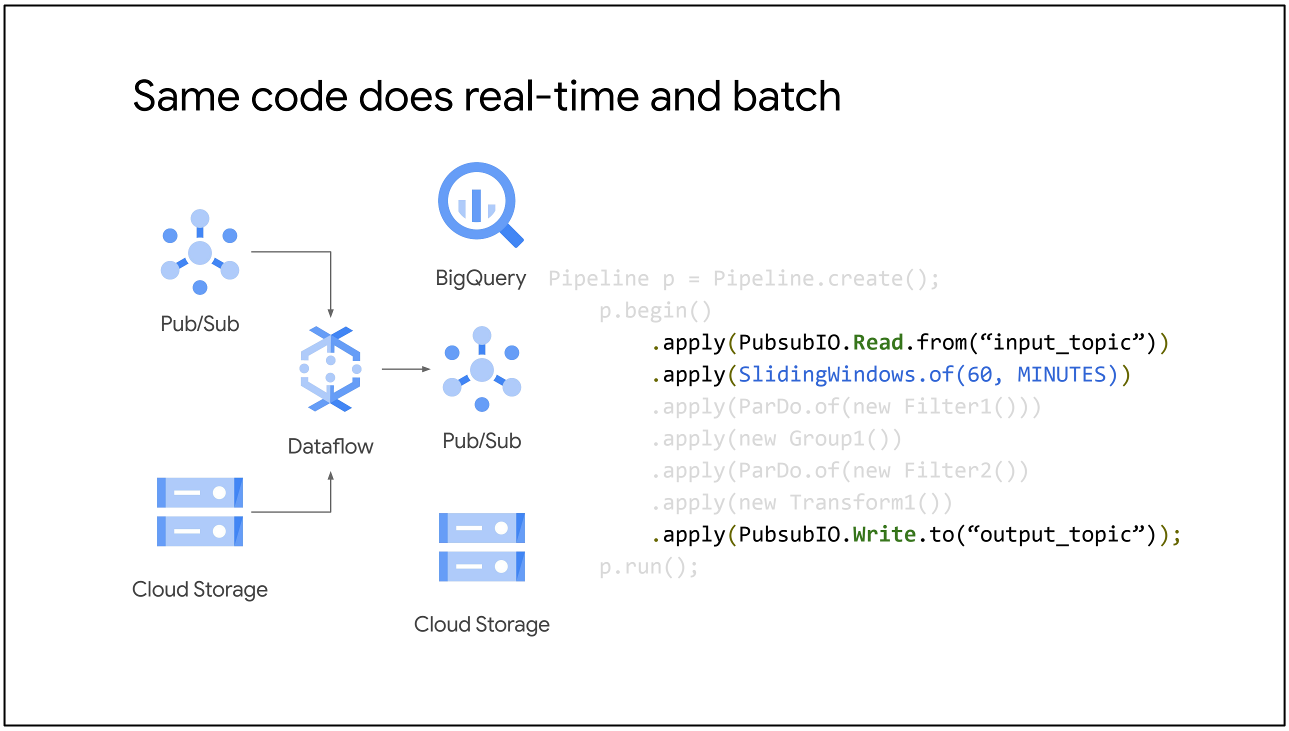
```
Pipeline p = Pipeline.create();
p
  .apply(TextIO.Read.from("gs://…
 "))
  .apply(ParDo.of(new Filter1()))
  .apply(new Group1())
 .apply(ParDo.of(new Filter2())
  .apply(new Transform1())
 .apply(TextIO.Write.to("gs://…")
 );
p.run();
```

```
class Filter1 extends DoFn<...> {
  public void
  processElement(ProcessContext c) {
      ... = c.element();
      ...
      c.output(...);
  }
```

Here’s what an actual pipeline looks like using the Apache Beam SDK on Cloud Dataflow

Each step of the pipeline does a filter, group, transform, compare, join, and so on. Transforms can be done in parallel which is how you get an elastic pipeline.

In the code snippet here, c.element() gets the input and c.output() sends the output to the next step of the pipeline.



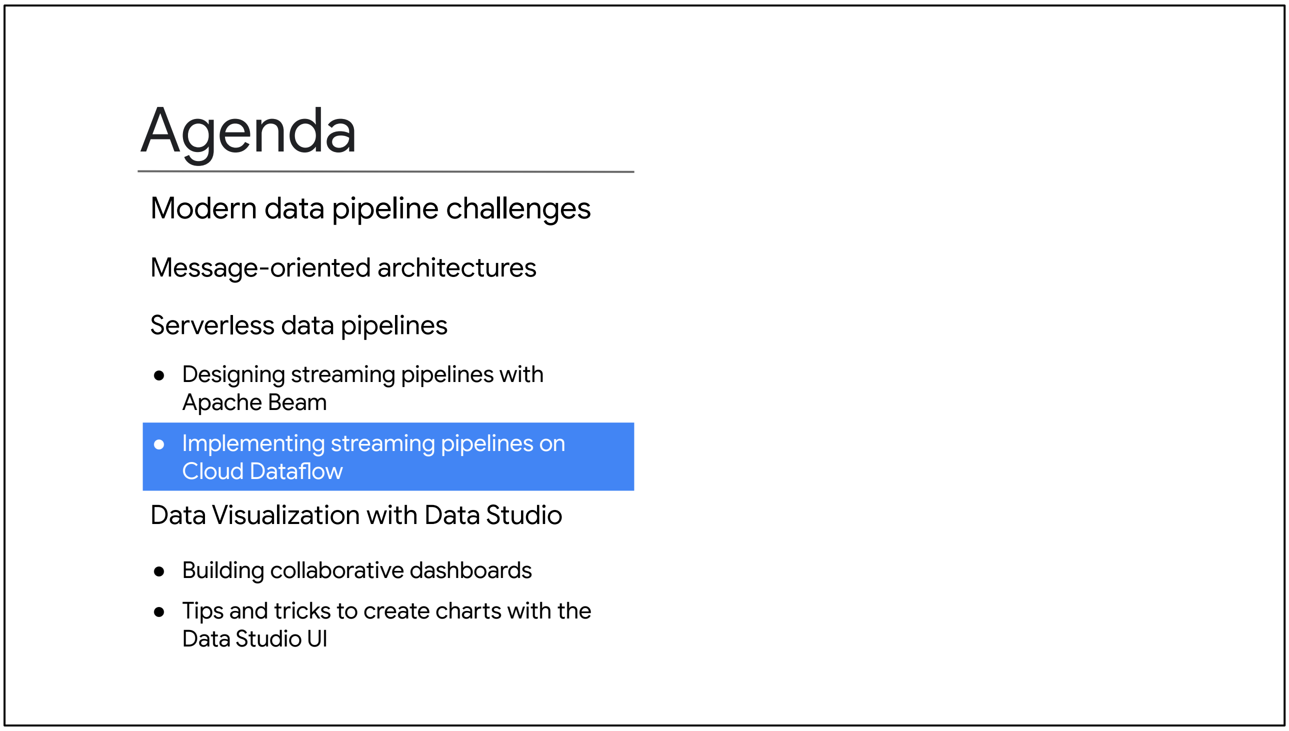
Same code does real-time and batch

Pub/Sub
BigQuery
Dataflow
Pub/Sub
Cloud Storage
Cloud Storage

```
Pipeline p = Pipeline.create();
    p.begin()
        .apply(PubsubIO.Read.from("input_topic"))
        .apply(SlidingWindows.of(60, MINUTES))
        .apply(ParDo.of(new Filter1()))
        .apply(new Group1())
        .apply(ParDo.of(new Filter2())
        .apply(new Transform1())
        .apply(PubsubIO.Write.to("output_topic"));
    p.run();
```

You can get input from any of several sources, and you can write output to any of several sinks. The pipeline code remains the same.

You can put this code inside a servlet, deploy it to App Engine, and schedule a cron task queue in App Engine to execute the pipeline periodically.



Agenda

Modern data pipeline challenges

Message-oriented architectures

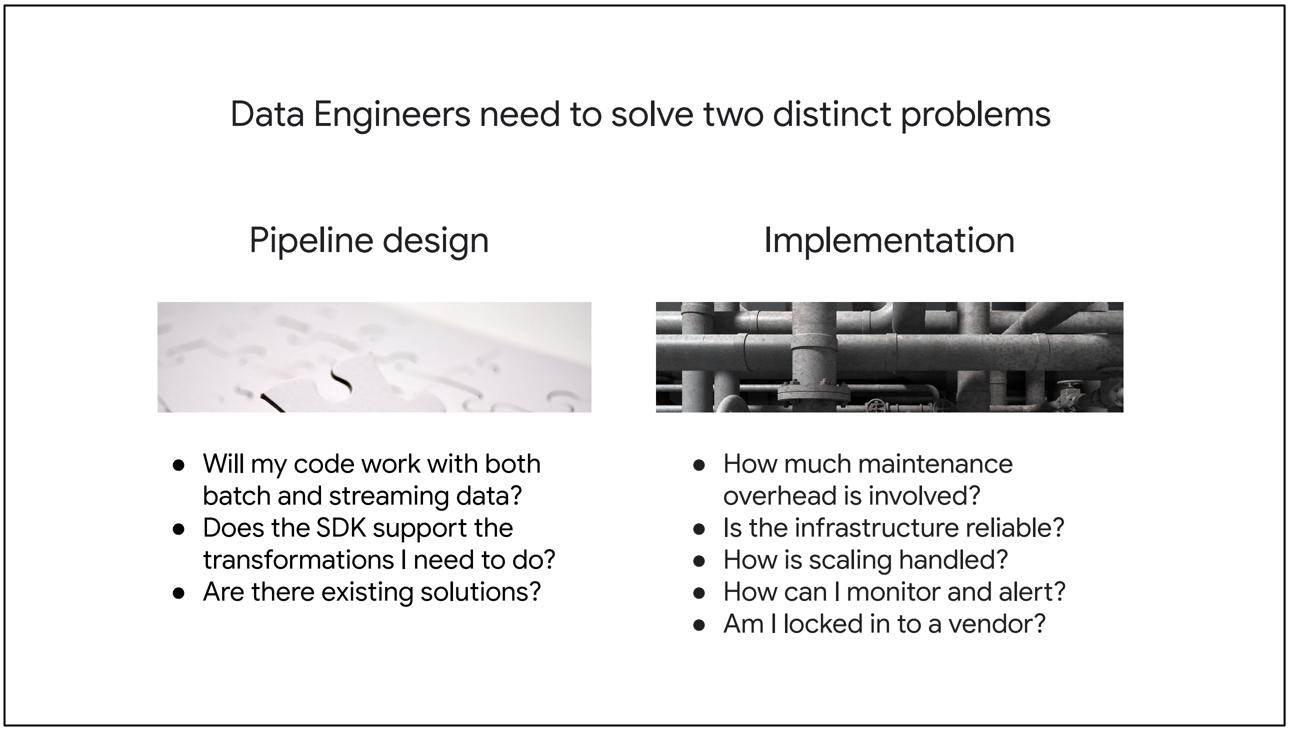
Serverless data pipelines

- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

Now that you've seen the pipeline design in code, it's time to talk about implementing it at scale in your production data pipelines.



Data Engineers need to solve two distinct problems

Pipeline design

- Will my code work with both batch and streaming data?
- Does the SDK support the transformations I need to do?
- Are there existing solutions?

Implementation

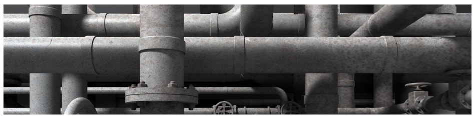
- How much maintenance overhead is involved?
- Is the infrastructure reliable?
- How is scaling handled?
- How can I monitor and alert?
- Am I locked in to a vendor?

When shopping for execution engines for your pipeline code, consider these questions:

- How much maintenance overhead is involved?
- Is the infrastructure reliable?
- How is scaling handled?
- How can I monitor and alert?
- Am I locked in to a vendor?

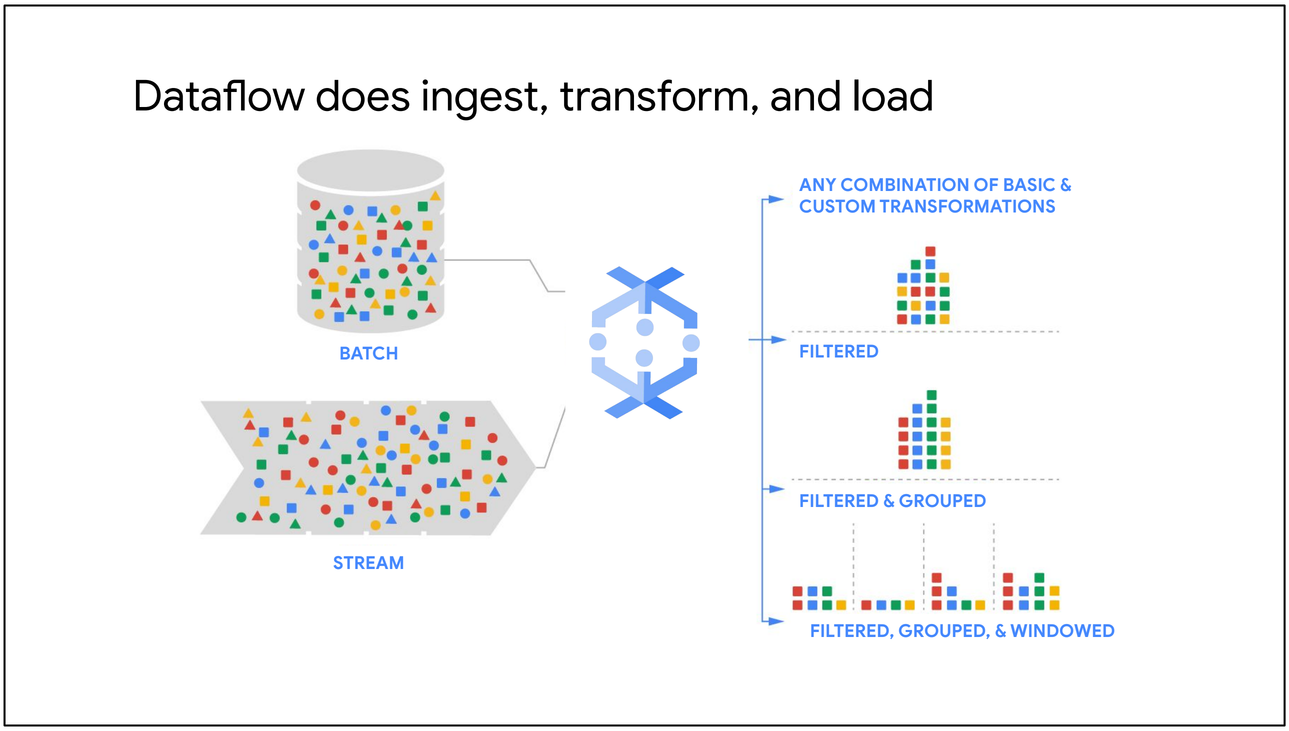
Data Engineers need to solve two distinct problems

Implementation with Dataflow

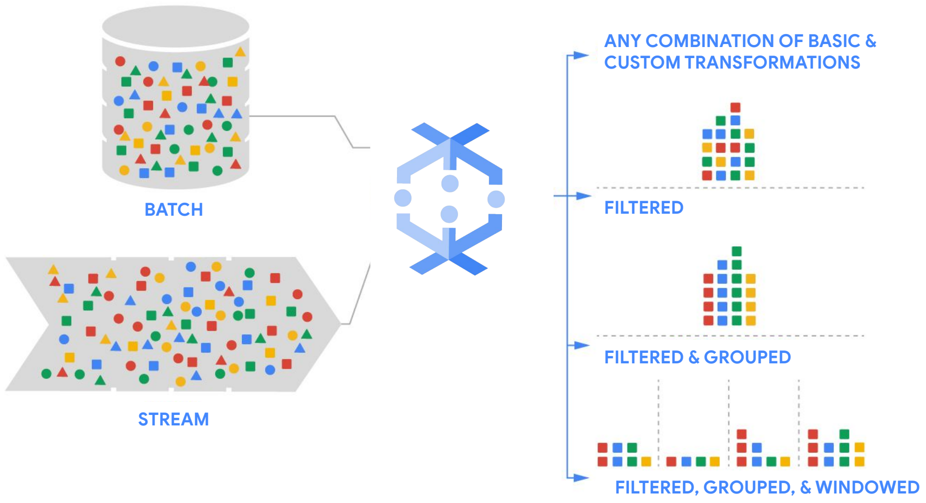


How much maintenance overhead is involved?	Little
Is the infrastructure reliable?	Built on Google infrastructure
How is scaling handled?	Autoscale workers
How can I monitor and alert?	Integrated with Operations
Am I locked in to a vendor?	Run Apache Beam elsewhere

With Cloud Dataflow a lot of the complexity of infrastructure setup and maintenance is handled for you. It’s built of Google infrastructure and autoscales to meet the demands of your data pipelines. It’s also tightly coupled with other Google Cloud Platform projects like Stackdriver so you can setup priority alerts and notifications to keep a watchful eye on your pipeline and the quality of data coming in and out.

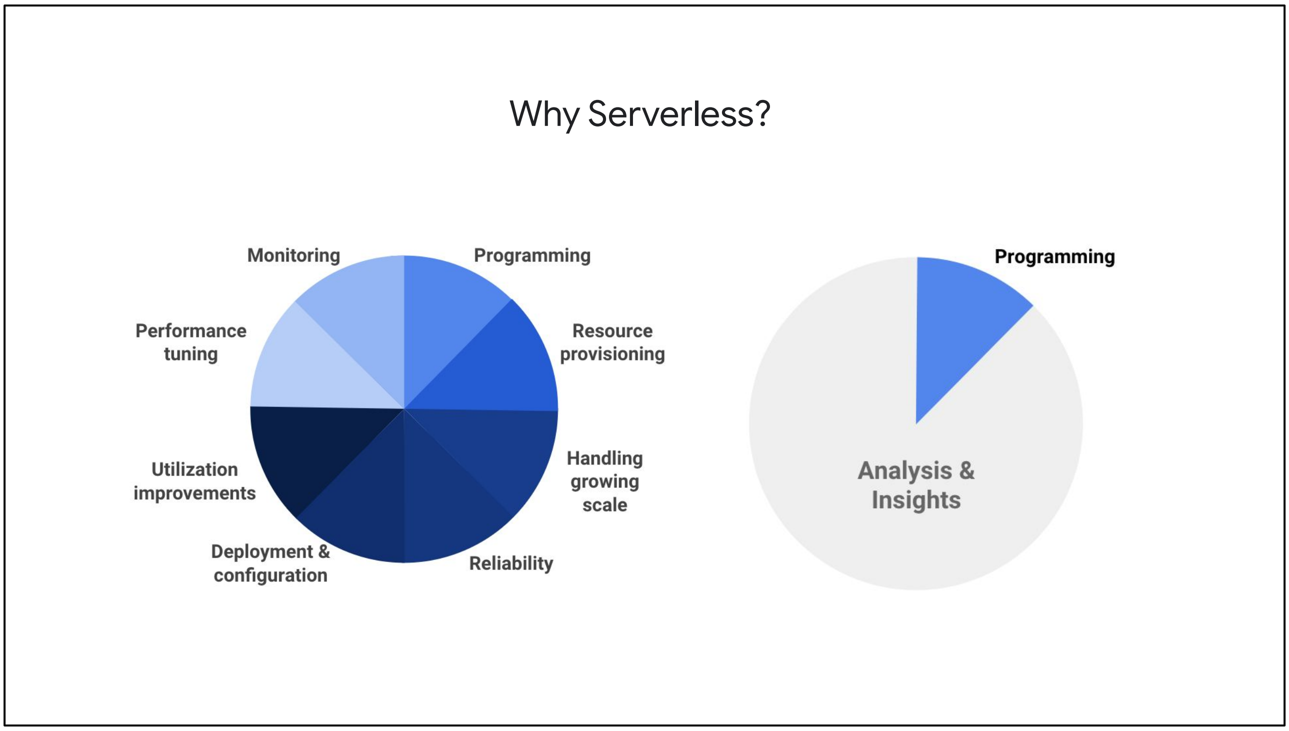


Dataflow does ingest, transform, and load


BATCH
STREAM
ANY COMBINATION OF BASIC & CUSTOM TRANSFORMATIONS
FILTERED
FILTERED & GROUPED
FILTERED, GROUPED, & WINDOWED

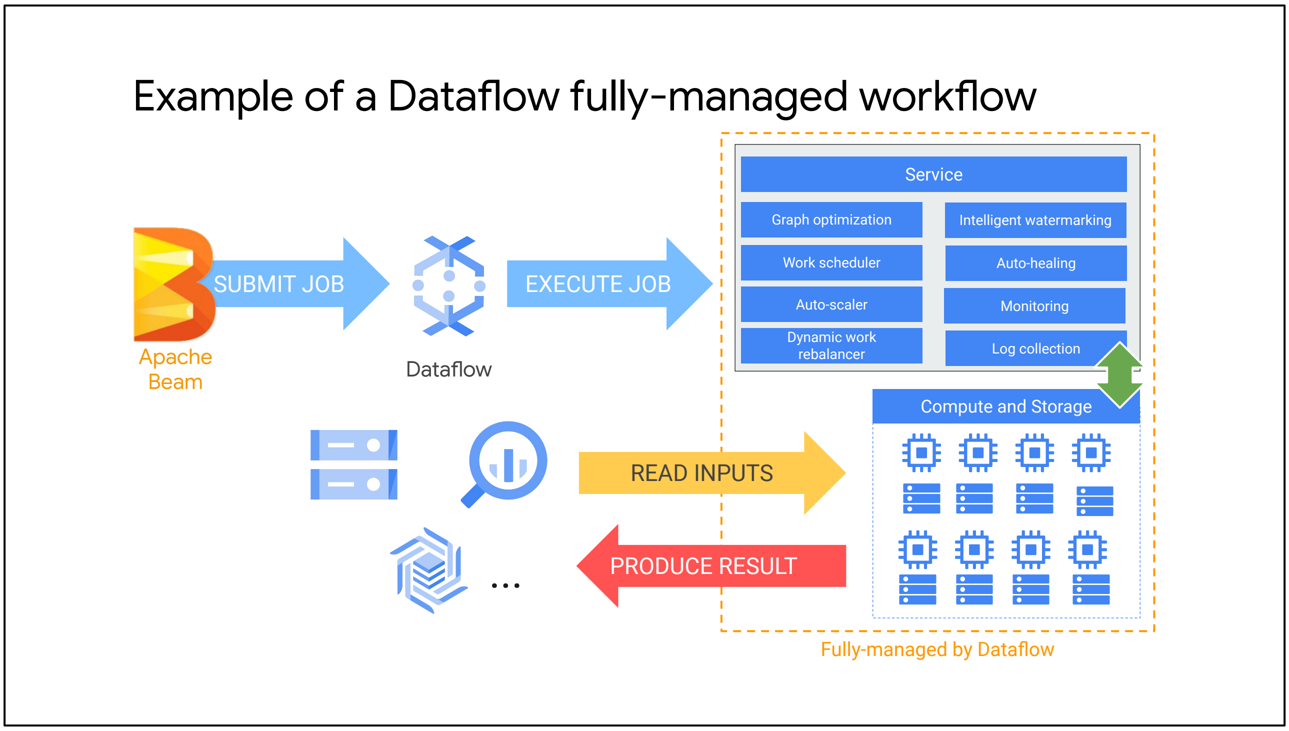
You can replace all the various data handling tools with just Dataflow. Many Hadoop workloads can be done easily and more maintainable with Dataflow. Plus, Dataflow is Serverless and NoOps.

What do we mean by serverless?

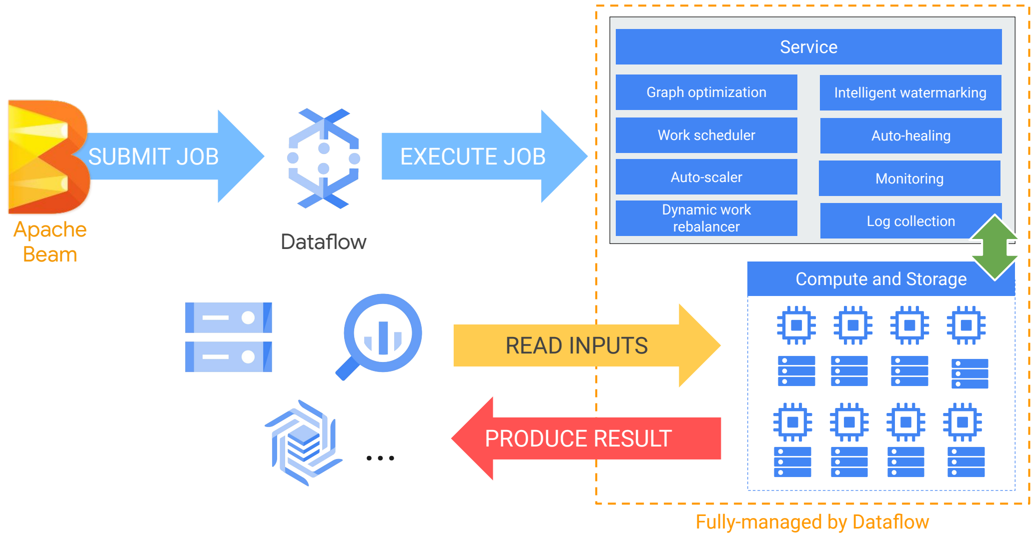


Why Serverless?

It means that Google will manage all the infrastructure tasks for you like resource provisioning and performance tuning as well as ensuring that your pipeline is reliable. That means you can spend more time analyzing the insights from your datasets and less time provisioning resources to ensure your pipeline will make it through it's next cycles. It's designed to be low maintenance.



Example of a Dataflow fully-managed workflow

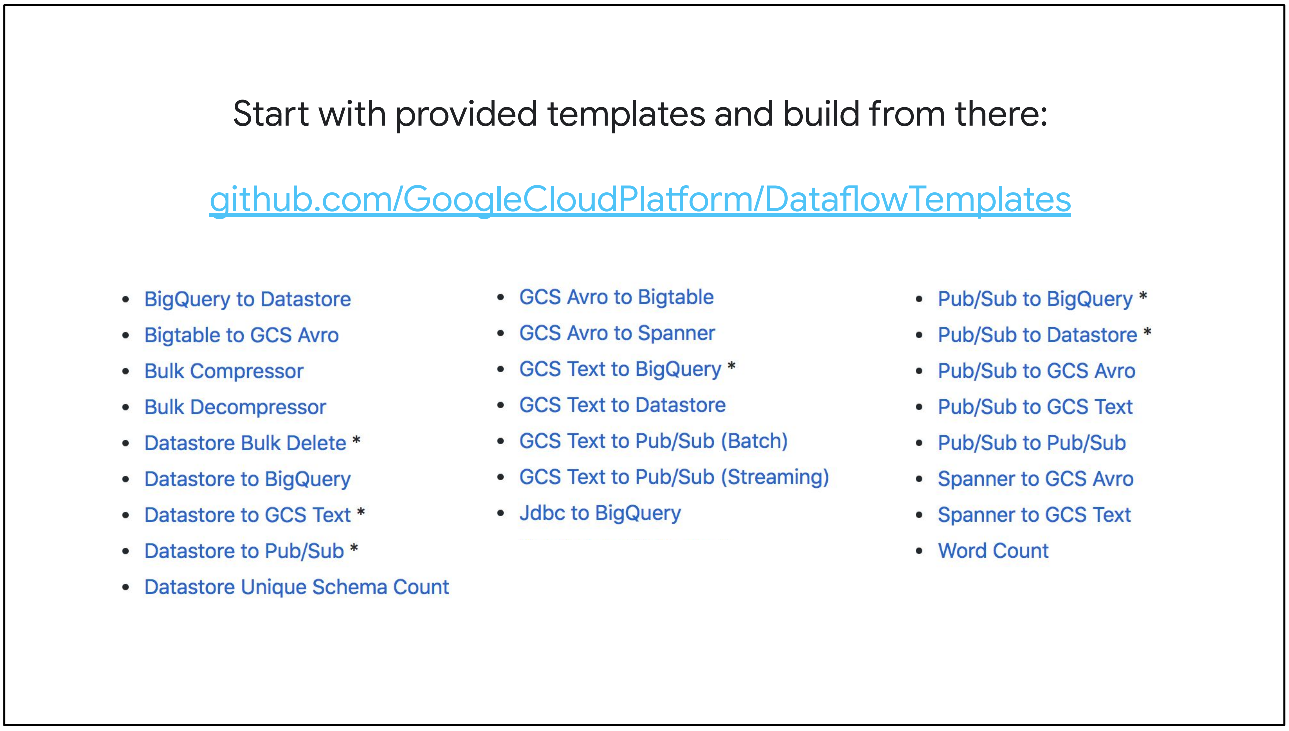


So what does Cloud Dataflow do with a job?

Once it receives it, the Cloud Dataflow service:

- Optimizes the execution graph of your model to remove inefficiencies
- It schedules out work in a distributed fashion to new workers and scales as needed
- It will auto-heal in the event of faults with workers
- It will re-balance automatically to best utilize it’s workers
- And it will connect to a variety of data sinks to product a result. BigQuery being one of many sinks that you can output data to as you will practice in your lab.

What you won’t see by design is all the Compute and Storage resources that Cloud Dataflow is managing for you elastically to fit the demand of your streaming data pipeline.



Start with provided templates and build from there:

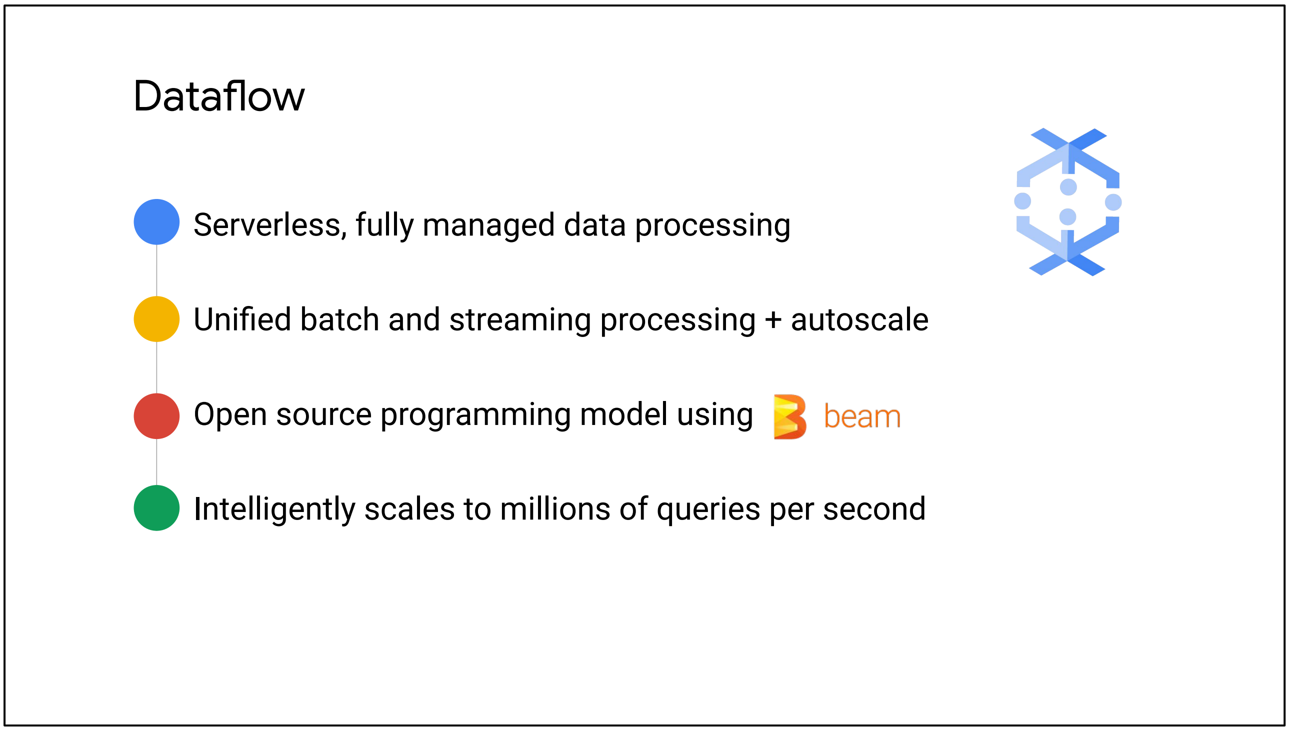
github.com/GoogleCloudPlatform/DataflowTemplates

- BigQuery to Datastore
- Bigtable to GCS Avro
- Bulk Compressor
- Bulk Decompressor
- Datastore Bulk Delete *
- Datastore to BigQuery
- Datastore to GCS Text *
- Datastore to Pub/Sub *
- Datastore Unique Schema Count
- GCS Avro to Bigtable
- GCS Avro to Spanner
- GCS Text to BigQuery *
- GCS Text to Datastore
- GCS Text to Pub/Sub (Batch)
- GCS Text to Pub/Sub (Streaming)
- Jdbc to BigQuery
- Pub/Sub to BigQuery *
- Pub/Sub to Datastore *
- Pub/Sub to GCS Avro
- Pub/Sub to GCS Text
- Pub/Sub to Pub/Sub
- Spanner to GCS Avro
- Spanner to GCS Text
- Word Count

Even if you're a seasoned Java or Python developer, it's still a good idea to start from many of the existing Dataflow templates that cover common use cases across Google Cloud Platform products.

See whether there is an existing template for you to work from. For IoT use cases, **Pub/Sub to BigQery** already exists and is a popular choice.

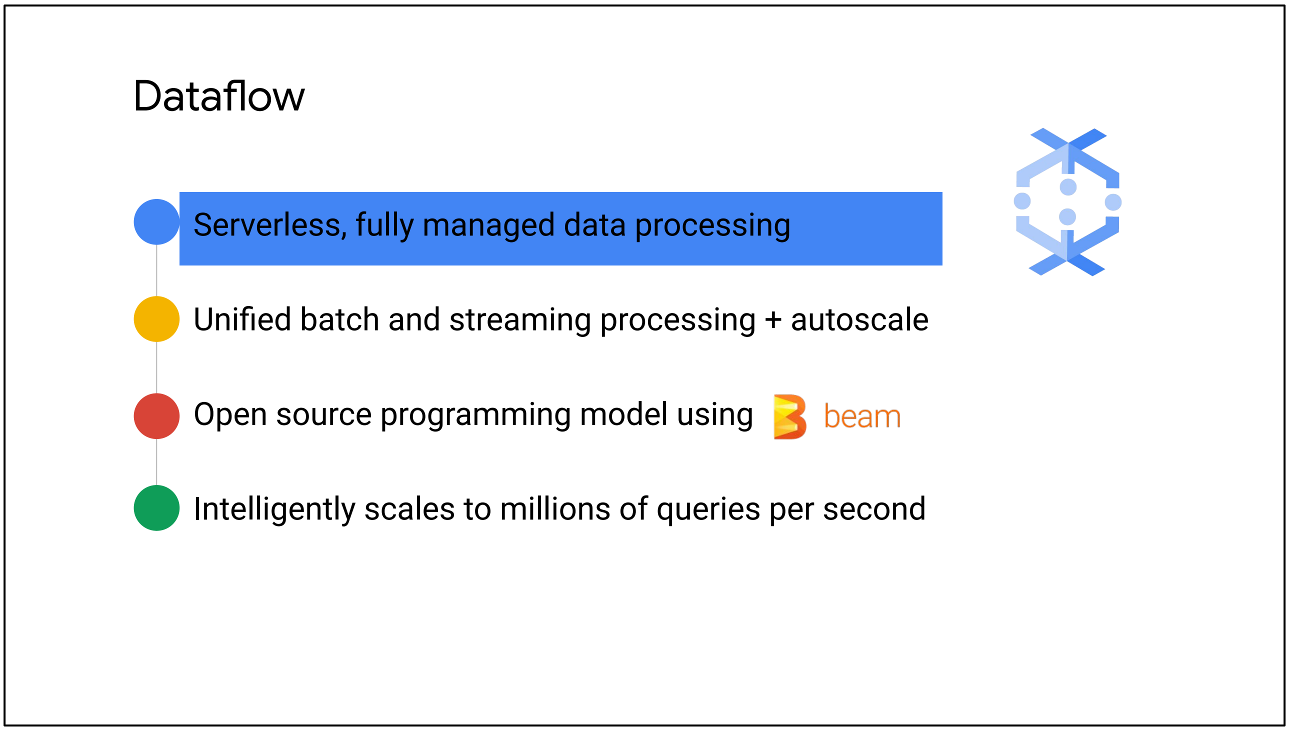
Github repo:
https://github.com/GoogleCloudPlatform/DataflowTemplates/blob/master/src/main/java/com/google/cloud/teleport/templates/PubSubToBigQuery.java



Dataflow

- Serverless, fully managed data processing
- Unified batch and streaming processing + autoscale
- Open source programming model using beam
- Intelligently scales to millions of queries per second

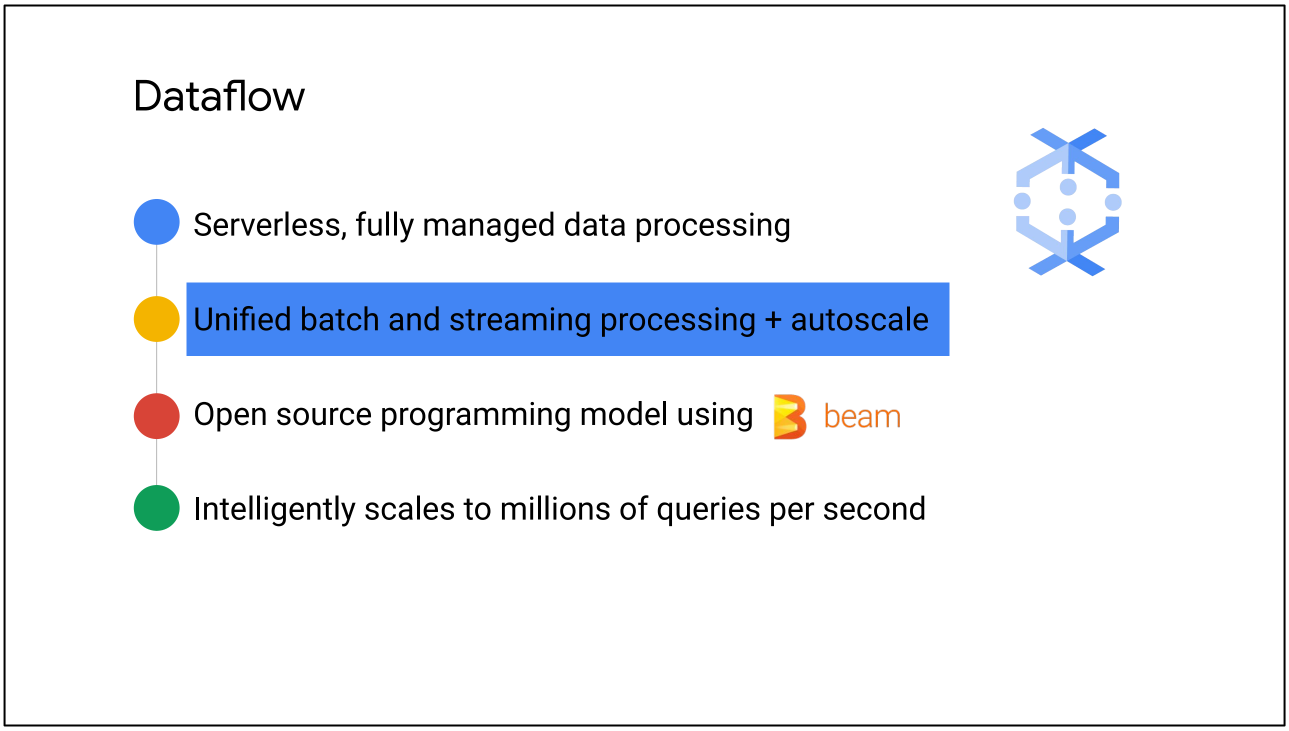
Here’s a quick recap of Cloud Dataflow



Dataflow

- Serverless, fully managed data processing
- Unified batch and streaming processing + autoscale
- Open source programming model using beam
- Intelligently scales to millions of queries per second

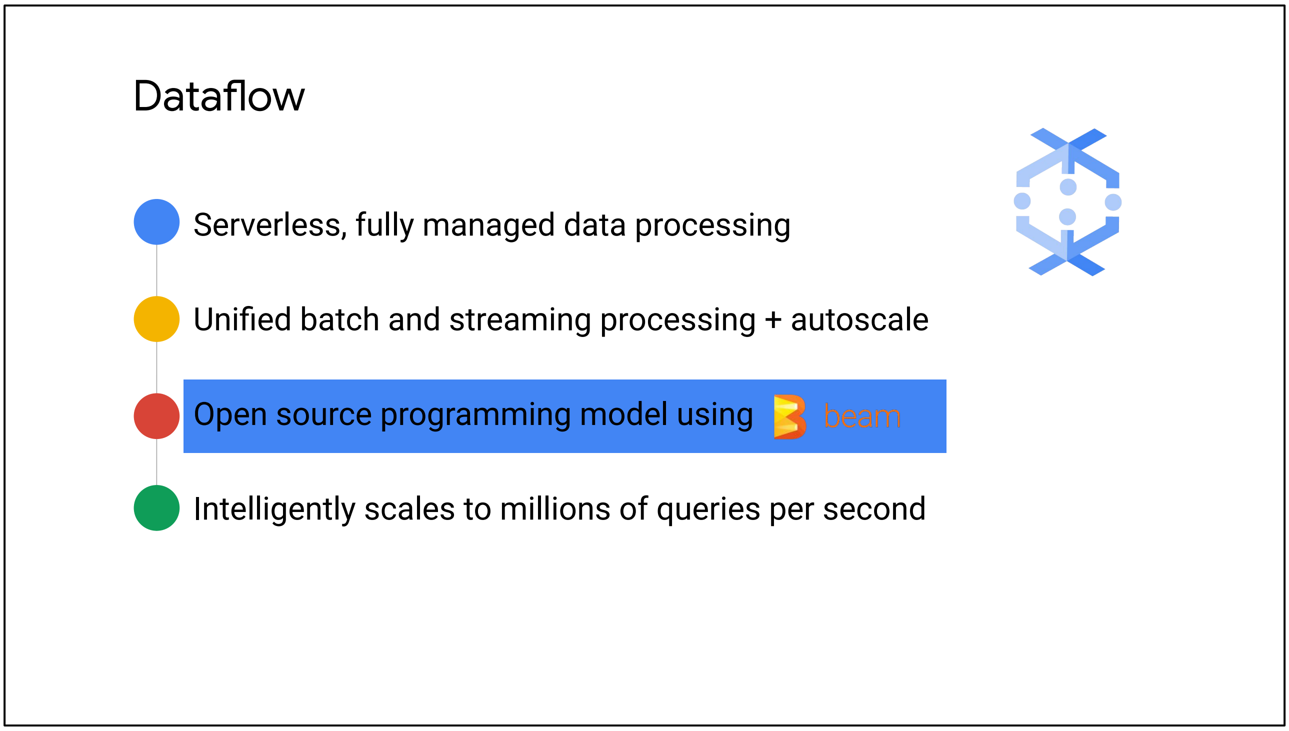
It’s designed to be easy to deploy your pipeline and have it just run. It’s serverless and fully managed



Dataflow

- Serverless, fully managed data processing
- Unified batch and streaming processing + autoscale
- Open source programming model using beam
- Intelligently scales to millions of queries per second

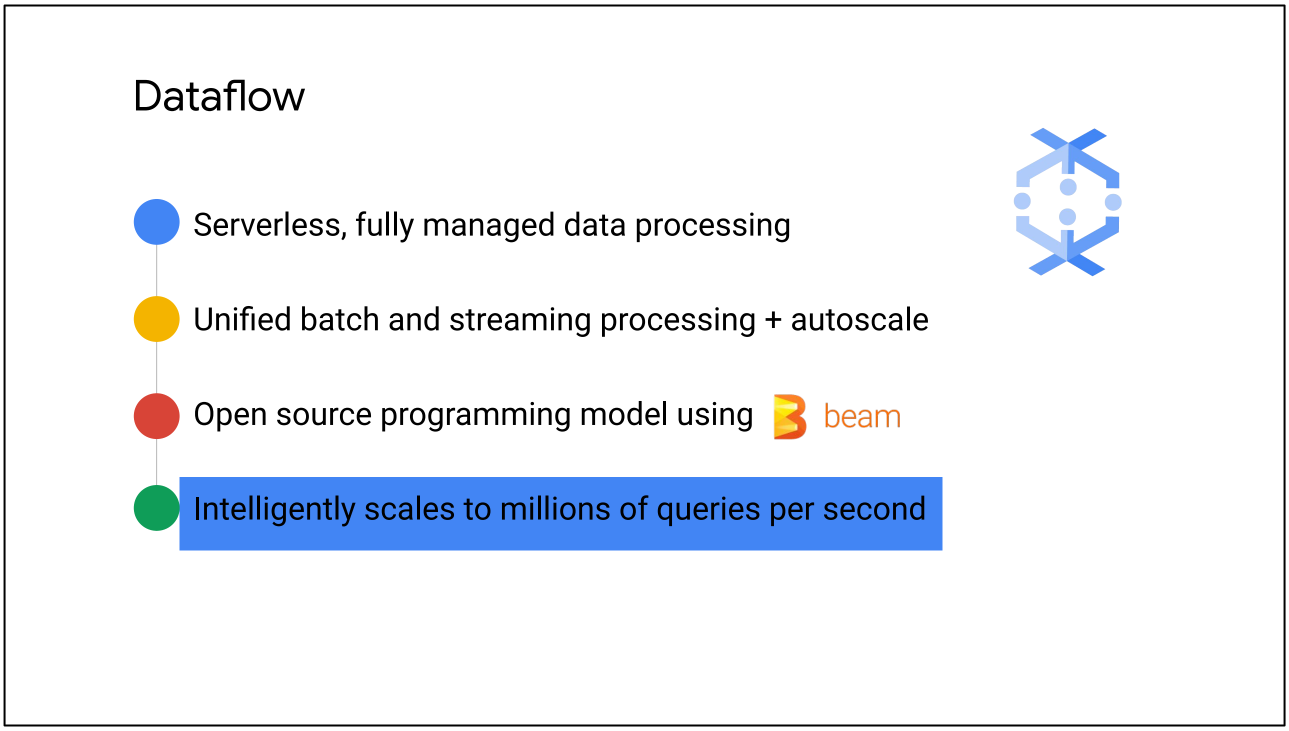
The same code can do both batch and streaming at scale



Dataflow

- Serverless, fully managed data processing
- Unified batch and streaming processing + autoscale
- Open source programming model using beam
- Intelligently scales to millions of queries per second

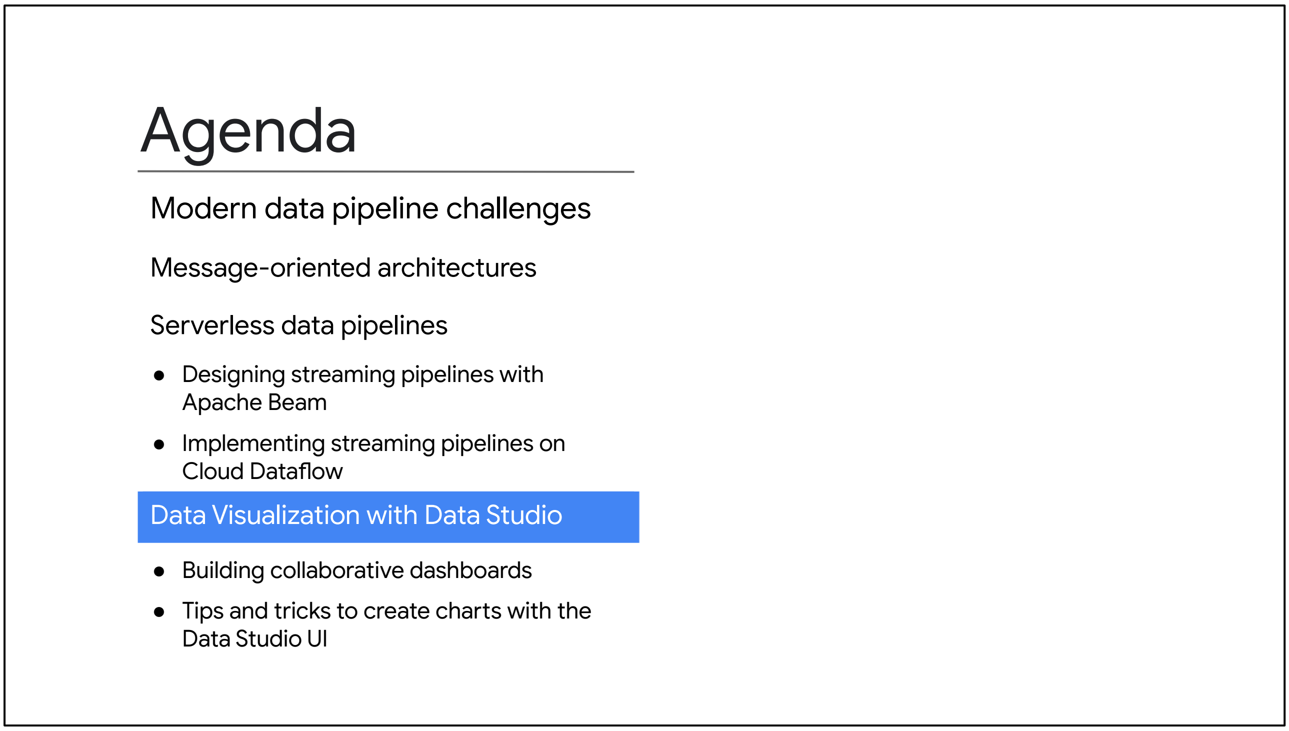
It’s built on Apache Beam’s open source programming model



Dataflow

- ● Serverless, fully managed data processing
- ● Unified batch and streaming processing + autoscale
- ● Open source programming model using beam
- ● Intelligently scales to millions of queries per second

And it can autoscale, heal, and re-balance to millions of queries per second



Agenda

Modern data pipeline challenges

Message-oriented architectures

Serverless data pipelines

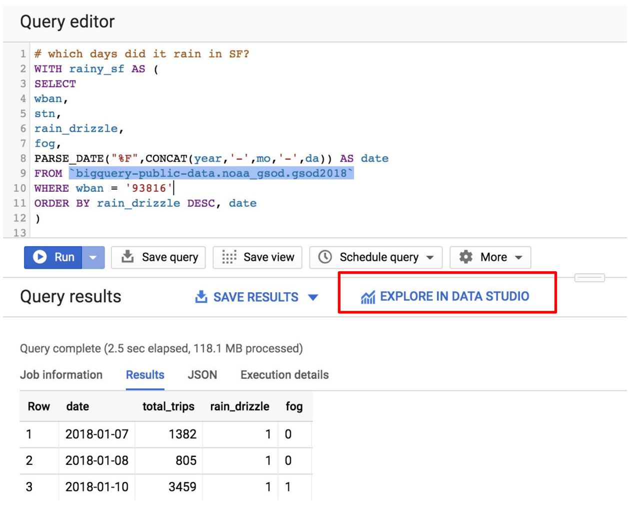
- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

While data sitting in BigQuery is great if you're an analyst exploring for insights -- it's often not enough for your stakeholders who want to see and interact with your data in a more intuitive and visual way. Data visualization is also a powerful tool for you to explore your data before presenting the results.

Explore Data Studio insights right from within BigQuery



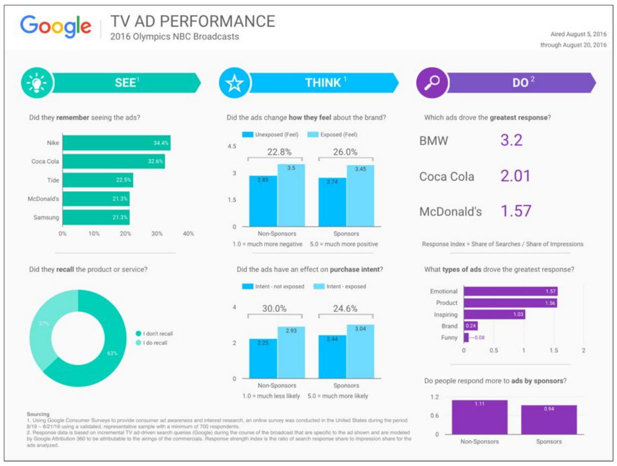
```
# which days did it rain in SF?
WITH rainy_sf AS (
SELECT
wban,
stn,
rain_drizzle,
fog,
PARSE_DATE("%F",CONCAT(year,'-',mo,'-',da)) AS date
FROM `bigquery-public-data.noaa_gsod.gsod2018`
WHERE wban = '93816'
ORDER BY rain_drizzle DESC, date
)
```

Query complete (2.5 sec elapsed, 118.1 MB processed)

Row	date	total_trips	rain_drizzle	fog
1	2018-01-07	1382	1	0
2	2018-01-08	805	1	0
3	2018-01-10	3459	1	1

Speaking of exploration, a new and popular way to explore your datasets is available right within BigQuery. After you execute a query you can Explore in Data Studio to immediately start creating visuals as part of a dashboard.

Build, collaborate, and share your dashboards



TV AD PERFORMANCE
2016 Olympics NBC Broadcasts
Aired August 5, 2016 through August 20, 2016
SEE[1]
THINK[1]
DO[2]
Did they remember seeing the ads?
Nike 34.4%
Coca Cola 32.6%
Tide 22.5%
McDonald's 21.3%
Samsung 21.3%
Did the ads change how they feel about the brand?
Unexposed (Feel)
Exposed (Feel)
22.8%
26.0%
Non-Sponsors 2.85 3.5
Sponsors 2.74 3.45
1.0 = much more negative 5.0 = much more positive
Which ads drove the greatest response?
BMW 3.2
Coca Cola 2.01
McDonald's 1.57
Response Index = Share of Searches / Share of Impressions
Did they recall the product or service?
I don't recall
I do recall
37%
63%
Did the ads have an effect on purchase intent?
Intent - not exposed
Intent - exposed
30.0%
24.6%
Non-Sponsors 2.25 2.93
Sponsors 2.44 3.04
1.0 = much less likely 5.0 = much more likely
What types of ads drove the greatest response?
Emotional 1.57
Product 1.56
Inspiring 1.03
Brand 0.24
Funny 0.08
Do people respond more to ads by sponsors?
Non-Sponsors 1.11
Sponsors 0.94

Sourcing
1. Using Google Consumer Surveys to provide consumer ad awareness and interest research, an online survey was conducted in the United States during the period 8/19 – 8/21/16 using a validated, representative sample with a minimum of 700 respondents.
2. Response data is based on incremental TV ad-driven search queries (Google) during the course of the broadcast that are specific to the ad shown and are modeled by Google Attribution 360 to be attributable to the airings of the commercials. Response strength index is the ratio of search response share to impression share for the ads analyzed.

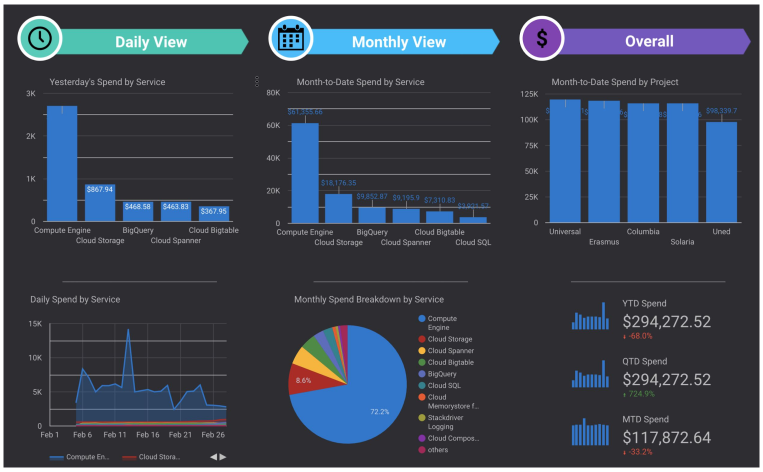
Tell a clear story with your data

Share and collaborate on reports with others

Here's an example of an actual Data Studio report. You don't have to understand the numbers, but just on the surface, what is this report trying to do?

In this case it was the Ad performance for the 2016 Olympic game broadcasted on NBC. The role of your report should be to highlight key insights that your audience cares about -- don’t show them everything, focus and filter their attention to just what data they need to see. Telling a good story with your data on a dashboard is critical because your users likely aren’t going to care about the amazing pipeline you just built if the output is a dashboard that is hard to use.

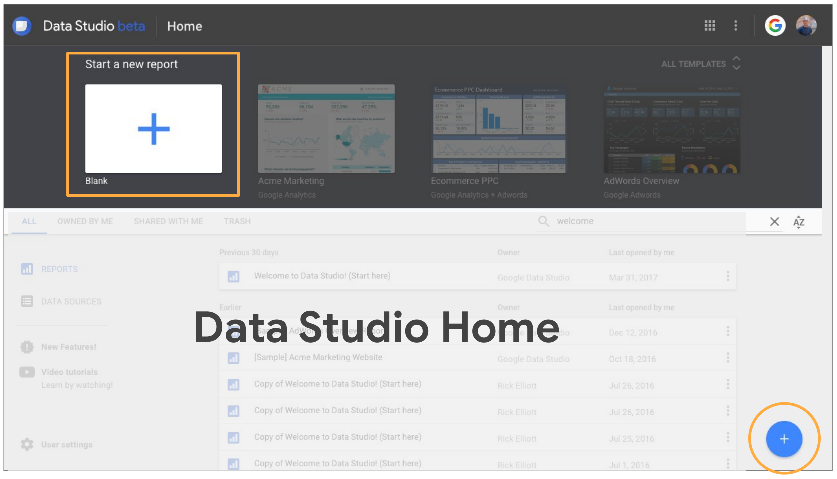
Access templates like this GCP Billing Dashboard



Daily View
Monthly View
Overall
Yesterday's Spend by Service
Month-to-Date Spend by Service
Month-to-Date Spend by Project
Daily Spend by Service
Monthly Spend Breakdown by Service
YTD Spend
$294,272.52
QTD Spend
$294,272.52
MTD Spend
$117,872.64

Data Studio comes with prebuilt templates like this one that monitors your GCP Billing in a dashboard. You can even connect this one to your GCP account to monitor your own spending and do some fun analysis like track incoming BigQuery jobs and data processed.

Create new reports in the Data Studio UI



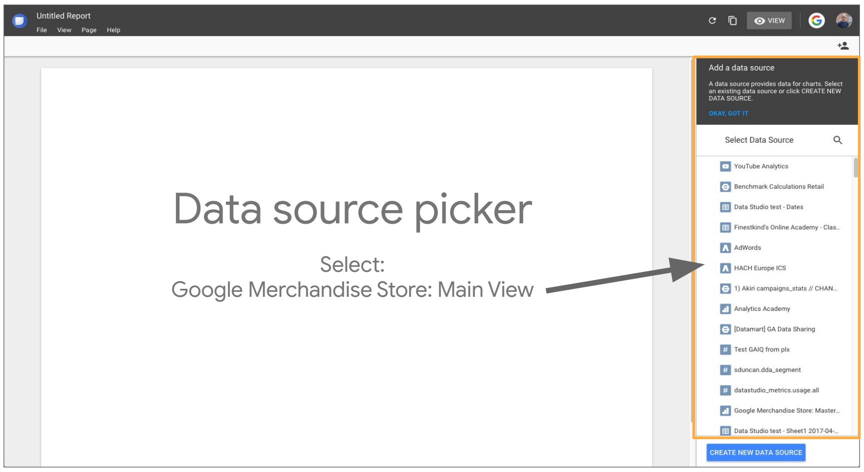
Here’s a quick walkthrough of the Data Studio UI. Keep in mind the team is continually updating the product so refer to their documentation and examples for additional practice.

The home page shows the dashboards and data sources you have access to. It’s an important distinction -- connected data sources can feed into dashboards but just because someone has access to your dashboard doesn’t mean they have permission to view the data presented (because that could be controlled in BigQuery or your GCP project).

Anyway, there two ways to create a new report from scratch: from the templates panel on top

Or from the button in the lower right.

Select data sources to build your visualizations

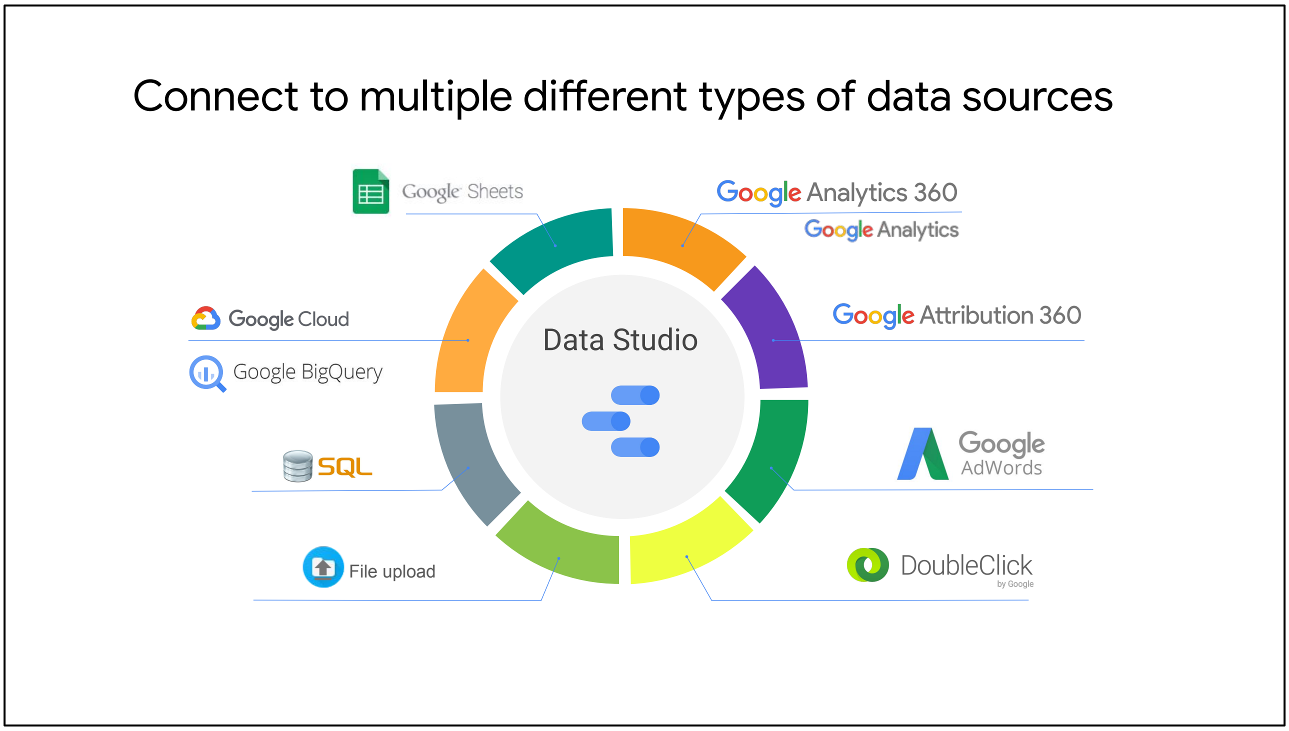

Untitled Report
File
View
Page
Help
VIEW
Data source picker
Select:
Google Merchandise Store: Main View
Add a data source
OKAY, GOT IT
Select Data Source
YouTube Analytics
Benchmark Calculations Retail
Data Studio test - Dates
Finestkind's Online Academy - Clas...
AdWords
HACH Europe ICS
1) Akiri campaigns_stats // CHAN...
Analytics Academy
[Datamart] GA Data Sharing
Test GAIQ from plx
sduncan.dda_segment
datastudio_metrics.usage.all
Google Merchandise Store: Master...
Data Studio test - Sheet1 2017-04-...
CREATE NEW DATA SOURCE

A data source provides data for charts. Select an existing data source or click CREATE NEW DATA SOURCE.

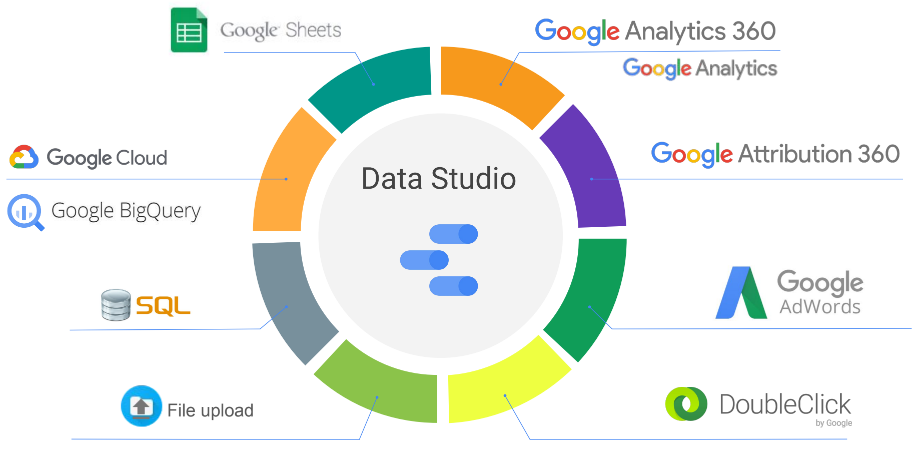
The first thing you need to do is tell Data Studio where your data is coming from. That is known as the "data source"

A Data Studio report can have any number of data sources, but we'll just start with one.

The data source picker shows all the data sources you have access to.



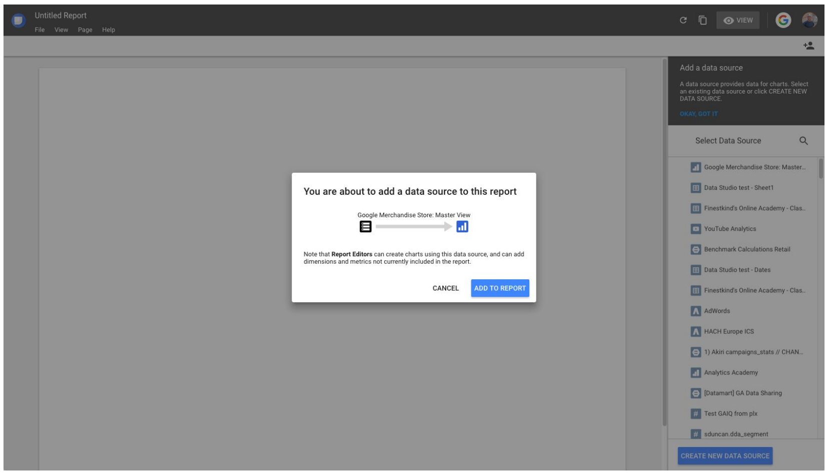
Connect to multiple different types of data sources



Note that you can have any or all of these data sources in a single Data Studio report.

Since Data Studio reports and datasets can be shared, you should be aware of the ramifications of adding a data source.

Add the data source to your report



You are about to add a data source to this report

Google Merchandise Store: Master View

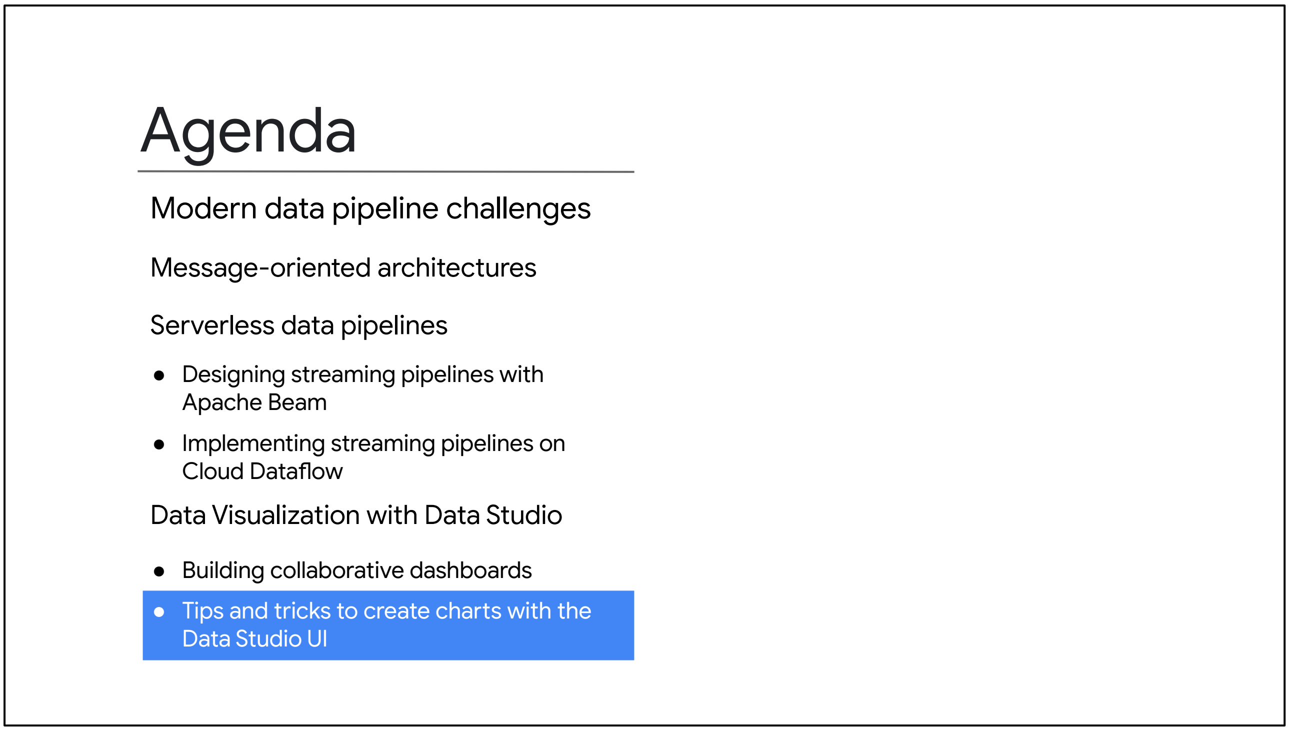
Note that **Report Editors** can create charts using this data source, and can add dimensions and metrics not currently included in the report.

A data source provides data for charts. Select an existing data source or click CREATE NEW DATA SOURCE.

- Google Merchandise Store: Master...
- Data Studio test - Sheet1
- Finestkind's Online Academy - Clas...
- YouTube Analytics
- Benchmark Calculations Retail
- Data Studio test - Dates
- Finestkind's Online Academy - Clas...
- AdWords
- HACH Europe ICS
- 1) Akiri campaigns_stats // CHAN...
- Analytics Academy
- [Datamart] GA Data Sharing
- Test GAIQ from plx
- sduncan.dda_segment

When you add a data source to a report, other people who can view the report can potentially see all the data in that data source if you share that source with them. And anyone who can edit the report can use all the fields from any added data sources to create new charts with them.

You can click ADD TO REPORT to connect the data source and then you are ready to start visualizing.



Agenda

Modern data pipeline challenges

Message-oriented architectures

Serverless data pipelines

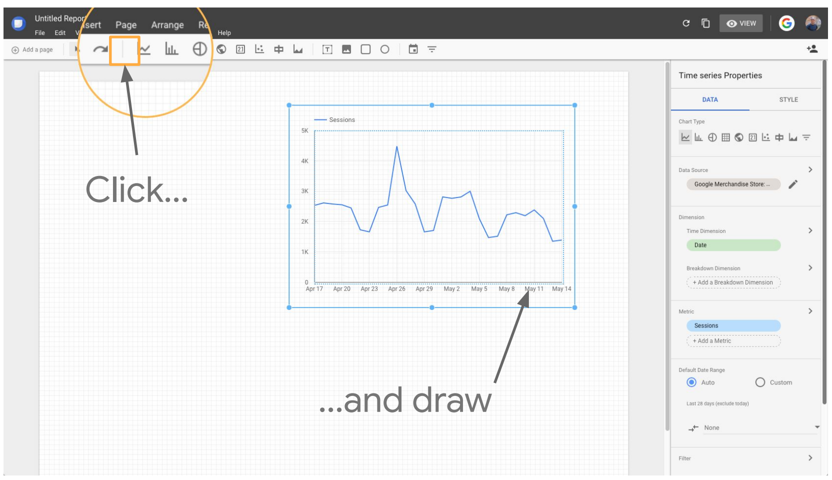
- Designing streaming pipelines with Apache Beam
- Implementing streaming pipelines on Cloud Dataflow

Data Visualization with Data Studio

- Building collaborative dashboards
- Tips and tricks to create charts with the Data Studio UI

Now let’s look at some tips and tricks for chart creation

Create charts to visualize data relationships



Time series Properties

DATA | STYLE

Chart Type

Data Source: Google Merchandise Store: ...

Dimension

Time Dimension: Date

Breakdown Dimension: + Add a Breakdown Dimension

Metric: Sessions

+ Add a Metric

Default Date Range: [✓ Auto] Custom

Last 28 days (exclude today)

None

Filter

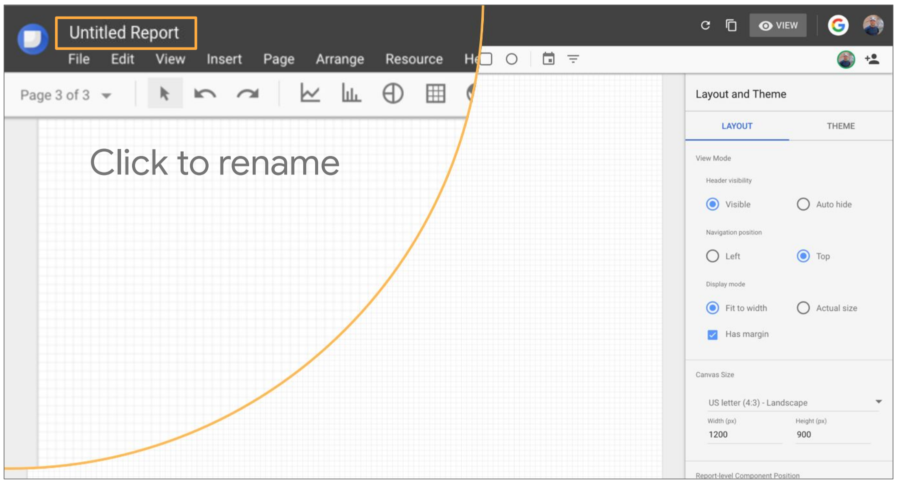
The mantra in Data Studio is "click and draw", not "drag and drop".

Select the chart type, then draw a box where you want it to go.

Data Studio automatically adds some appropriate fields to the chart. The chart properties panel on the right is where you configure the chart.

Add a time series to the report

Add a descriptive name to your report

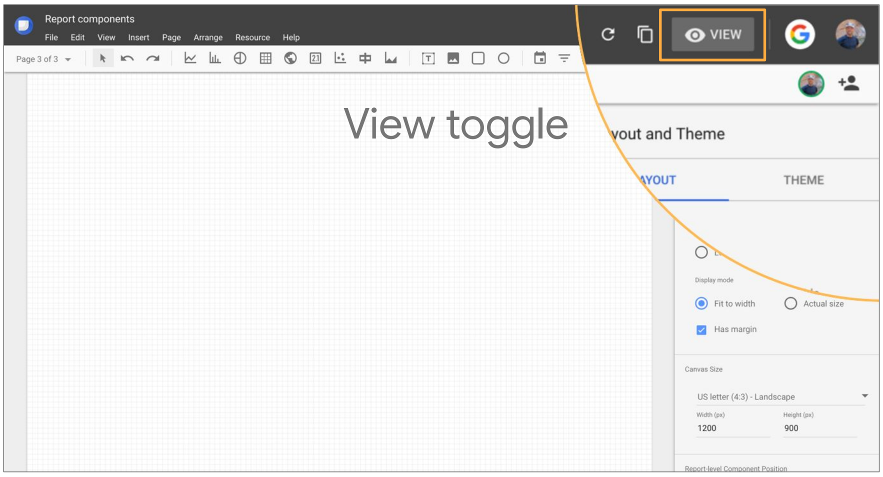


View Mode
Header visibility: [✓ Visible] Auto hide
Navigation position: Left [✓ Top]
Display mode: [✓ Fit to width] Actual size
[✓ Has margin]
Canvas Size: US letter (4:3) - Landscape
Width (px): 1200
Height (px): 900
Report-level Component Position

Give your report a name by clicking on the title.

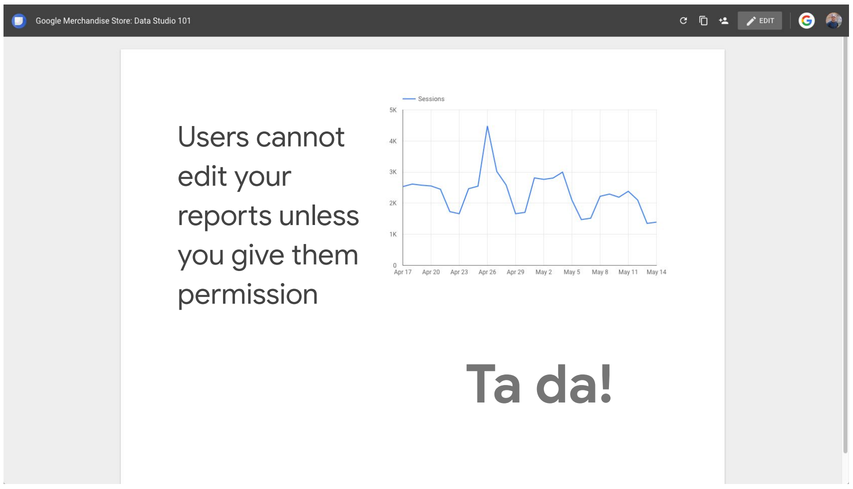
One thing to watch out for, since Data Studio is based on Drive, you can have duplicate file names.

View the end-user version of the report



Click View to see what your users will see

View your report as an end-user



Users cannot edit your reports unless you give them permission

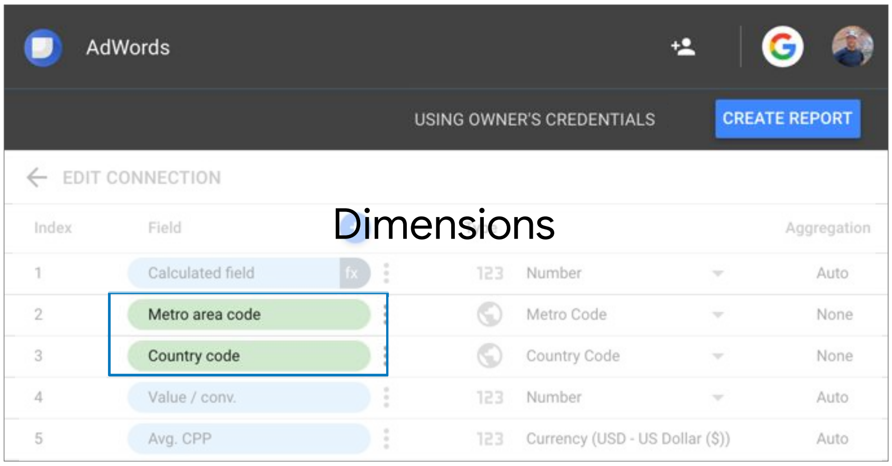
Ta da!

And here is your report. Notice it looks very similar to when you were editing it, but as a viewer, you can't modify the report.

Note that you can mouse over any charts and graphs on your report -- it is live data, not just a static image.

So this report may not look like much, but with just a few mouse clicks, you can connect a Google Analytics data source and created a chart graphing session data over time.

Understand the date source shade

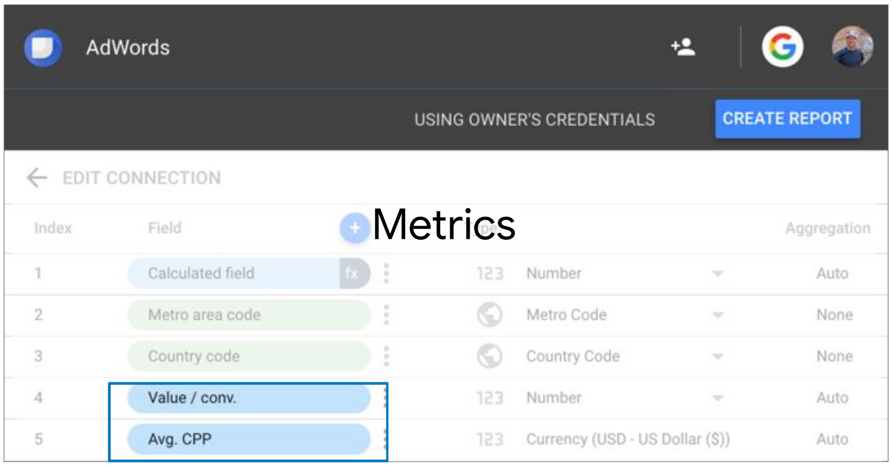


Index	Field		Aggregation
1	Calculated field	Number	Auto
2	Metro area code	Metro Code	None
3	Country code	Country Code	None
4	Value / conv.	Number	Auto
5	Avg. CPP	Currency (USD - US Dollar ($))	Auto

A few more key things to know.

Dimension chips are green. Dimensions are categories or buckets of information. Dimension values are names, descriptions or other characteristics of a category.

Understand the date source shade



AdWords

USING OWNER'S CREDENTIALS

CREATE REPORT

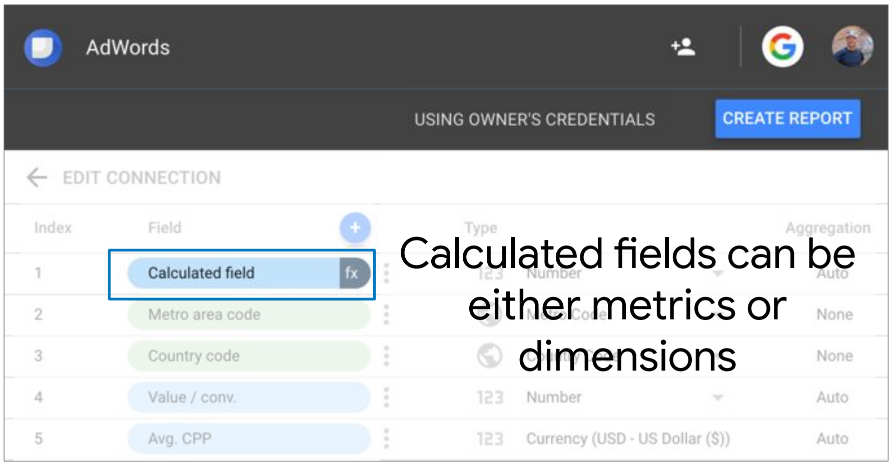
EDIT CONNECTION

Metrics

Index	Field	Type		Aggregation
1	Calculated field	123	Number	Auto
2	Metro area code		Metro Code	None
3	Country code		Country Code	None
4	Value / conv.	123	Number	Auto
5	Avg. CPP	123	Currency (USD - US Dollar ($))	Auto

Metric chips are blue. Metrics measure dimension values. Metrics represent measurements or aggregations, such as a sum, (X+Y), a count (how many X?) or ratio (X/Y).

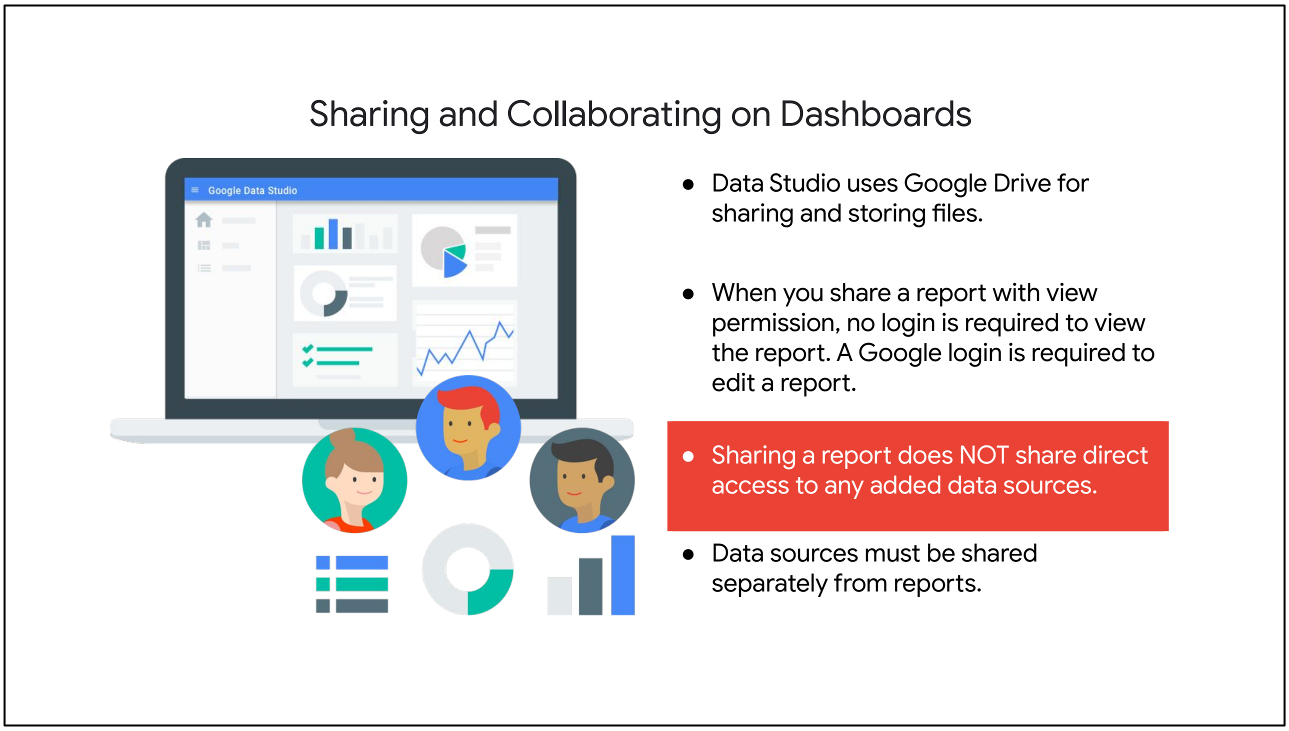
Understand the date source shade


AdWords
USING OWNER'S CREDENTIALS
CREATE REPORT
EDIT CONNECTION
Calculated fields can be either metrics or dimensions

Index	Field	Type	Aggregation
1	Calculated field	Number	Auto
2	Metro area code	Metro Code	None
3	Country code		None
4	Value / conv.	Number	Auto
5	Avg. CPP	Currency (USD - US Dollar ($))	Auto

You can create simple metrics, like field 1 plus field 2. These can be much more complex, using any of the over 50 calculated field functions available.

A calculated field could also be a dimension. For example, you can use a REGULAR EXPRESSION to extract part of a landing page URL or keyword into a new dimension to make it easier to read.



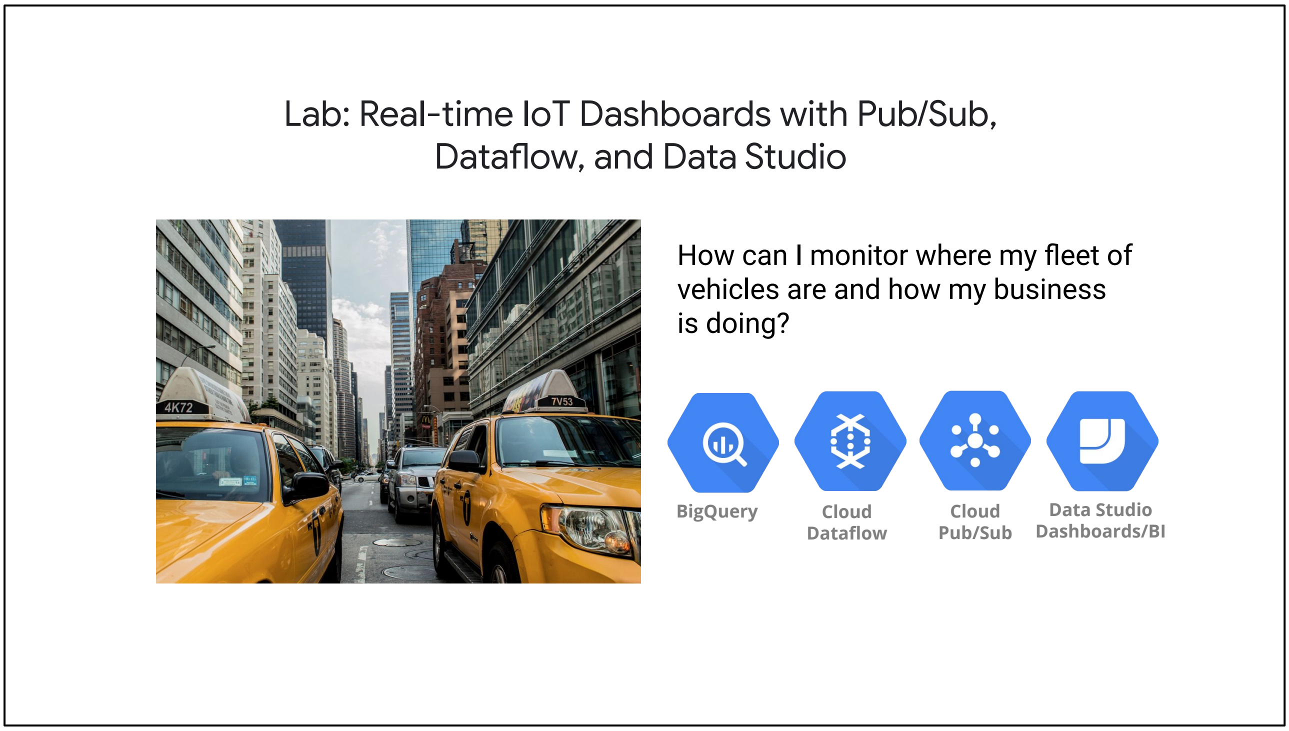
Sharing and Collaborating on Dashboards

- Data Studio uses Google Drive for sharing and storing files.
- When you share a report with view permission, no login is required to view the report. A Google login is required to edit a report.
- Sharing a report does NOT share direct access to any added data sources.
- Data sources must be shared separately from reports.

Lastly, you can share and collaborate on your dashboards with your team. Keep in mind that when you share a report, if you're connecting to an underlying datasource like BigQuery -- Data Studio does not grant permissions to viewers on that data source if the viewer doesn't have them. This is for data security reasons.

After you share your report, users can interact with filters and sorting and you can collect feedback on the usage of your report through Data Studio's integration with Google Analytics.

[resource]
Sharing Google Data Studio reports:
https://support.google.com/datastudio/answer/6287179?hl=en



Lab: Real-time IoT Dashboards with Pub/Sub, Dataflow, and Data Studio

How can I monitor where my fleet of vehicles are and how my business is doing?

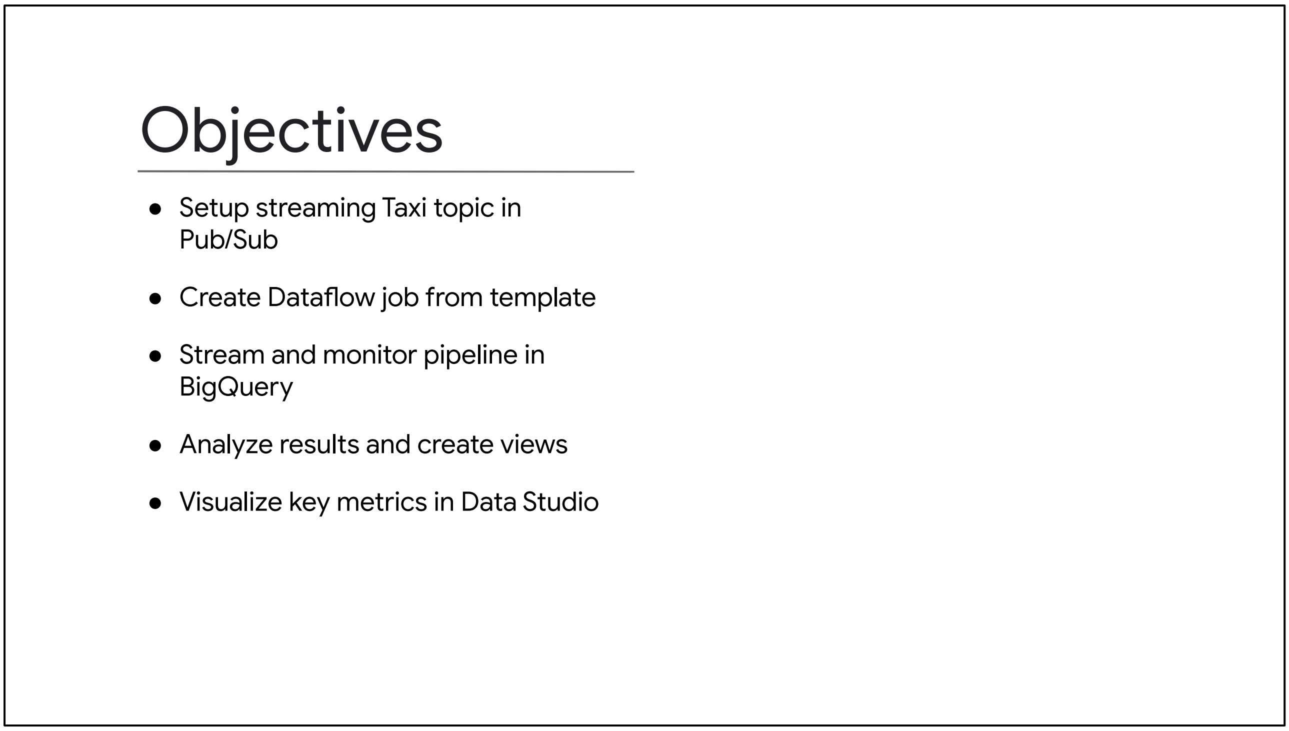
BigQuery

Cloud Dataflow

Cloud Pub/Sub

Data Studio Dashboards/BI

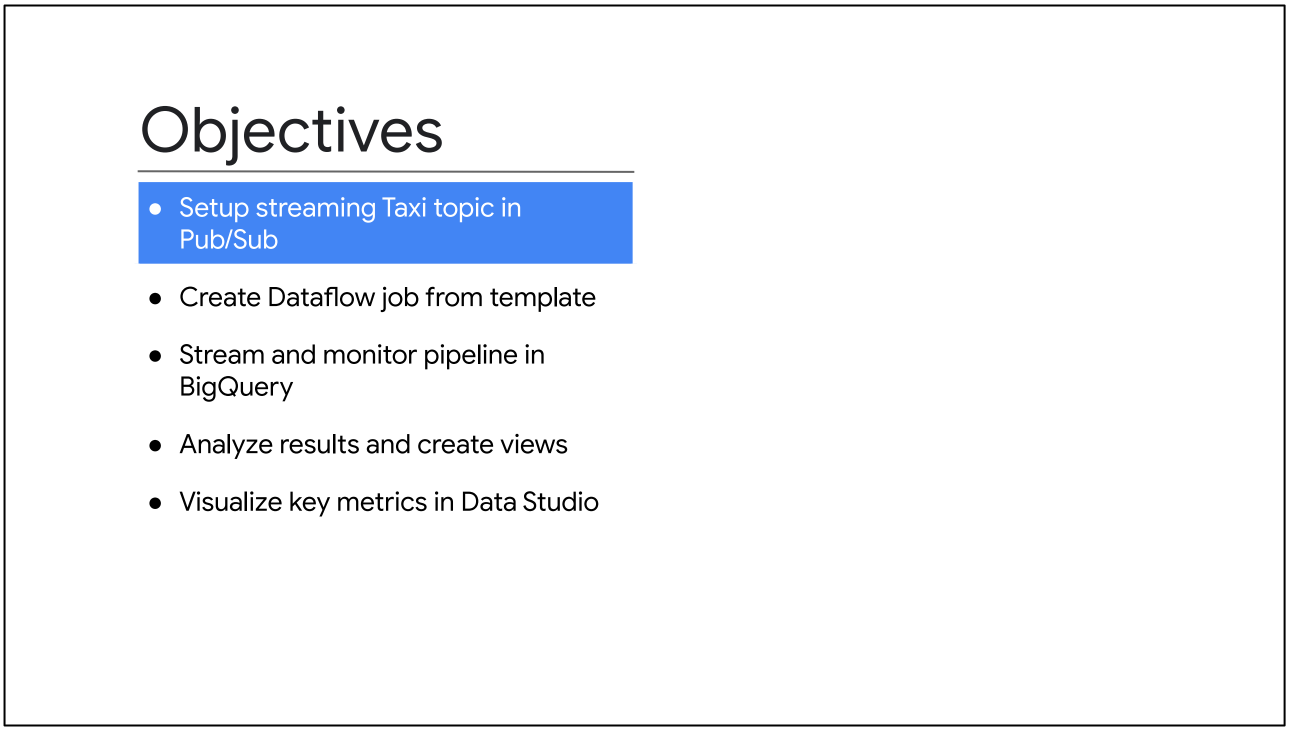
Now it's time for you to put your skills to the test with all the tools you just learned about. In this lab you will be building an end-to-end streaming data pipeline to monitor your sensor data and report on your dataset in near-real time on a dashboard.



Objectives

- Setup streaming Taxi topic in Pub/Sub
- Create Dataflow job from template
- Stream and monitor pipeline in BigQuery
- Analyze results and create views
- Visualize key metrics in Data Studio

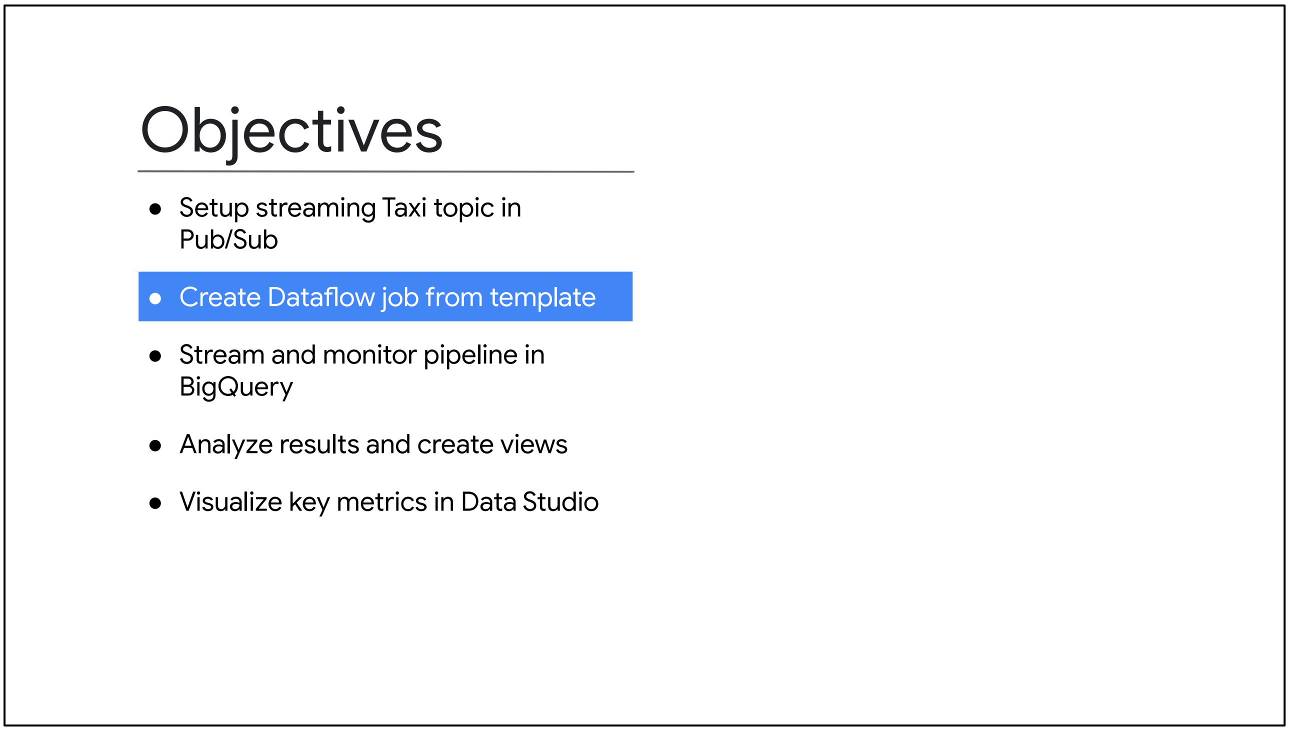
The objectives are:



Objectives

- Setup streaming Taxi topic in Pub/Sub
- Create Dataflow job from template
- Stream and monitor pipeline in BigQuery
- Analyze results and create views
- Visualize key metrics in Data Studio

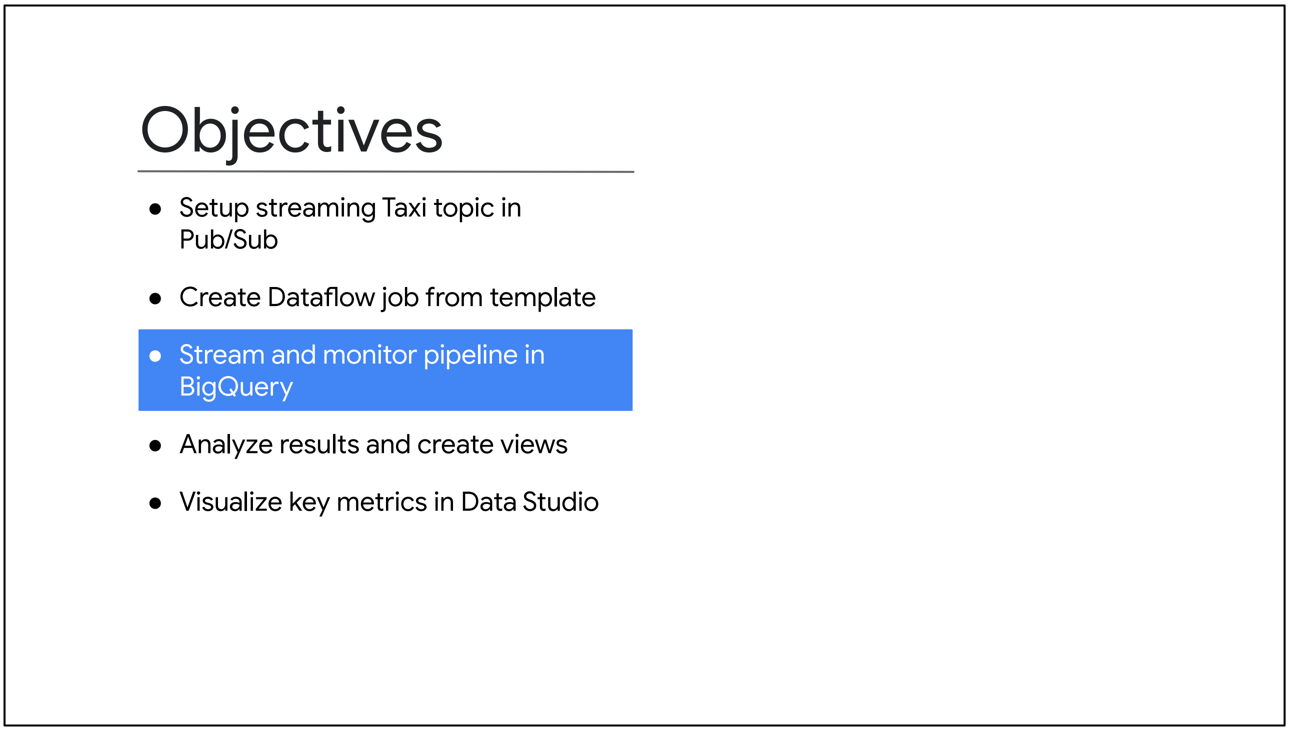
Setup a streaming topic in Pub/Sub for subscribers to listen to



Objectives

- Setup streaming Taxi topic in Pub/Sub
- Create Dataflow job from template
- Stream and monitor pipeline in BigQuery
- Analyze results and create views
- Visualize key metrics in Data Studio

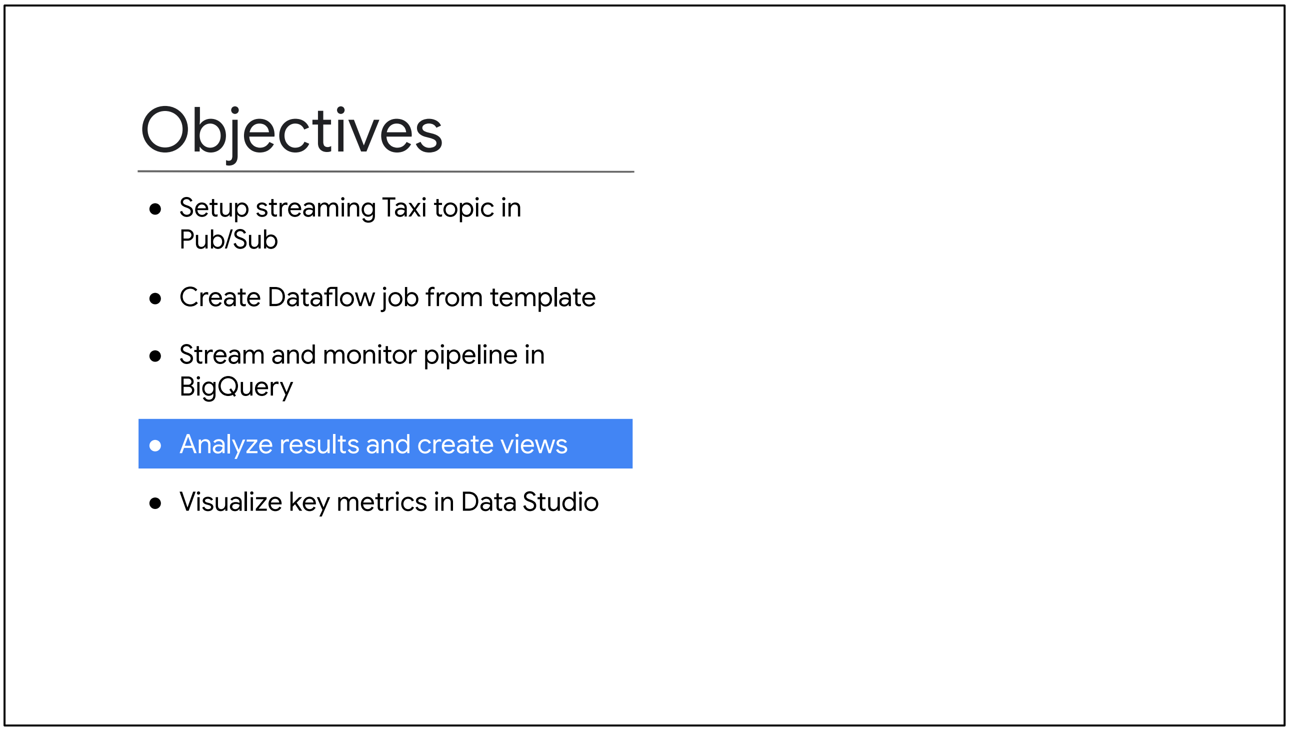
Create Dataflow job from a pre-existing template and subscribe to the pub/sub topic



Objectives

- Setup streaming Taxi topic in Pub/Sub
- Create Dataflow job from template
- Stream and monitor pipeline in BigQuery
- Analyze results and create views
- Visualize key metrics in Data Studio

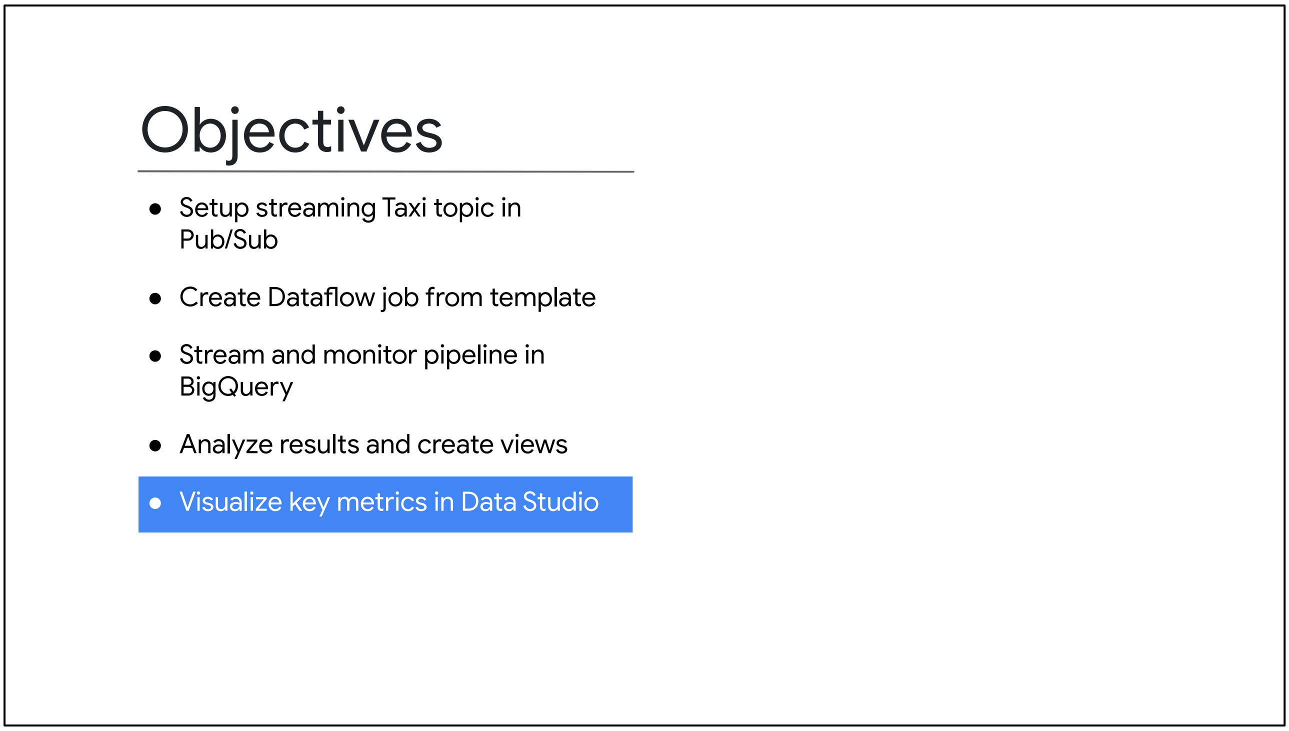
Stream and monitor your DataFlow pipeline into BigQuery



Objectives

- Setup streaming Taxi topic in Pub/Sub
- Create Dataflow job from template
- Stream and monitor pipeline in BigQuery
- **Analyze results and create views**
- Visualize key metrics in Data Studio

Analyze results with SQL and create views for visualization later



Objectives

- Setup streaming Taxi topic in Pub/Sub
- Create Dataflow job from template
- Stream and monitor pipeline in BigQuery
- Analyze results and create views
- Visualize key metrics in Data Studio

And then visualize key metrics in Data Studio

Keep in mind you have multiple attempts at each lab so if you don't get through it the first time or want to experiment with it more later you can come back and start a new instance.