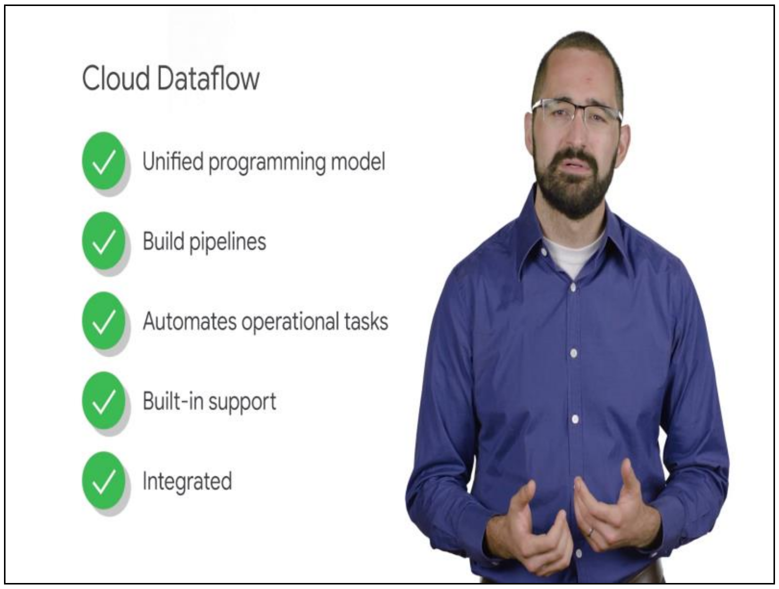
Google Cloud Computing Foundation Course

Evan Jones
Technical Curriculum Developer
Google Cloud

Lecture-72
Build ETL Pipelines using Cloud Dataflow

(Refer Slide Time: 00:10)



Cloud Dataflow

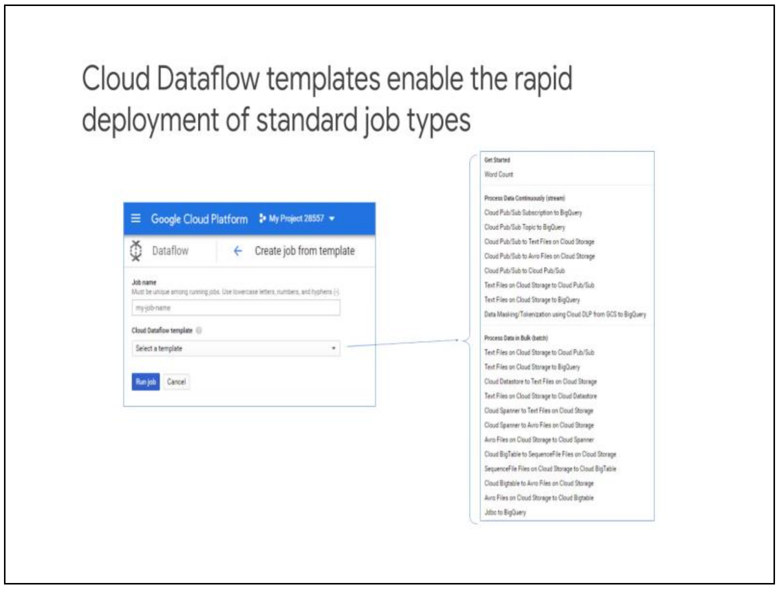
- ✓ Unified programming model
- ✓ Build pipelines
- ✓ Automates operational tasks
- ✓ Built-in support
- ✓ Integrated

In this topic you learn how you can use cloud dataflow to perform extract, transform and load operations. Cloud dataflow offers simplified streaming and batch data processing. It is a data processing service that is based on Apache beam that lets you develop and execute a range of data processing patterns, extract, transform and load batch and streaming. You use cloud dataflow to build data pipelines, monitor their execution and transform and analyze that data.

Importantly the same pipelines the same code that you are going to write works both for batch data and streaming data. You will explore pipelines more in details shortly. Cloud dataflow fully automates operational tasks like resource management and performance optimization for your pipeline. All resources are provided on-demand and automatically scale to meet requirements. Cloud dataflow provides built-in support for fault tolerant execution that is consistent and correct regardless of data size, cluster size, processing pattern or even the complexity of your pipeline.

Through its integration with the GCP console cloud dataflow provides statistics such as pipeline throughput and lag as well as the consolidated worker log and inspection all in near real-time. It also integrates with cloud storage, cloud pub/sub, cloud datastore, cloud BigTable and bigquery for seamless data processing. It is the glue that can hold it all together. Can also be extended to interact with other sources and syncs like Apache Kafka and HDFS.

(Refer Slide Time: 01:42)



Cloud Dataflow templates enable the rapid deployment of standard job types

Job name
Must be unique among running jobs. Use lowercase letters, numbers, and hyphens (-).
my-job-name

Cloud Dataflow template
Select a template

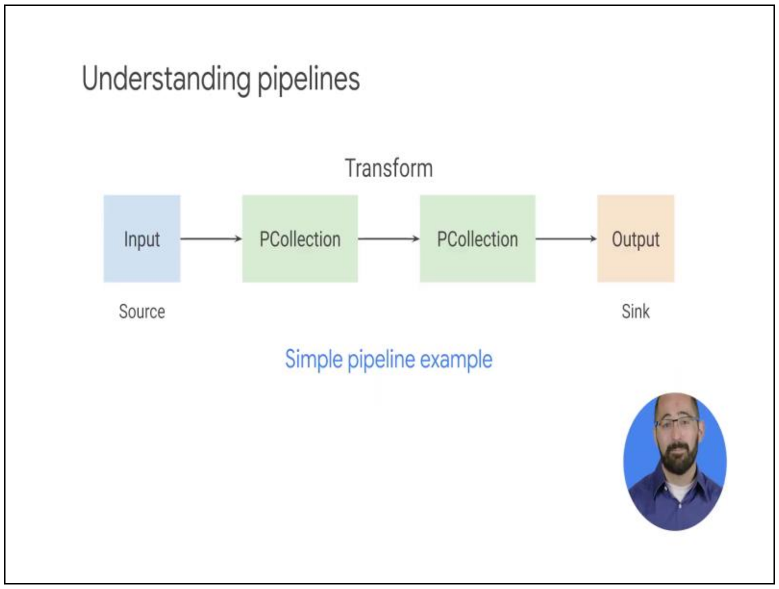
Get Started
Word Count

Process Data Continuously (stream)
Cloud Pub/Sub Subscription to BigQuery
Cloud Pub/Sub Topic to BigQuery
Cloud Pub/Sub to Text Files on Cloud Storage
Cloud Pub/Sub to Avro Files on Cloud Storage
Cloud Pub/Sub to Cloud Pub/Sub
Text Files on Cloud Storage to Cloud Pub/Sub
Text Files on Cloud Storage to BigQuery
Data Masking/Tokenization using Cloud DLP from GCS to BigQuery

Process Data in Bulk (batch)
Text Files on Cloud Storage to Cloud Pub/Sub
Text Files on Cloud Storage to BigQuery
Cloud Datastore to Text Files on Cloud Storage
Text Files on Cloud Storage to Cloud Datastore
Cloud Spanner to Text Files on Cloud Storage
Cloud Spanner to Avro Files on Cloud Storage
Avro Files on Cloud Storage to Cloud Spanner
Cloud BigTable to SequenceFile Files on Cloud Storage
SequenceFile Files on Cloud Storage to Cloud BigTable
Cloud Bigtable to Avro Files on Cloud Storage
Avro Files on Cloud Storage to Cloud Bigtable
Jdbc to BigQuery

Google provides QuickStart templates for cloud dataflow to allow you to rapidly deploy a number of useful data pipelines without requiring any Apache beam programming experience. The templates also remove the need to develop the pipeline code and therefore the need to consider the management of component dependencies in that pipeline code. You will do a lab later we will create a streaming pipeline using one of these Google Cloud dataflow templates. Let us look at pipelines now in more detail.

(Refer Slide Time: 02:15)



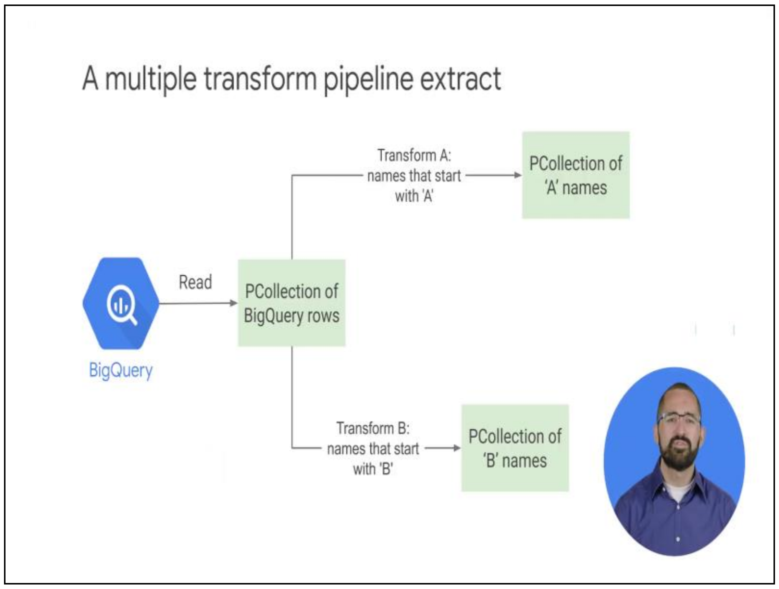
A pipeline represents a complete process on one or more data sets. The data can be brought in from external data sources it could then have a series of transformation operations such as filters, joins, aggregations etc applied to that data to give it some meaning and to achieve its desired form. This data could then be written to a sink. The sink could be within GCP or external the sink could be even the same as the data source.

The pipeline itself is what is called a directed acyclic graph or a dag, peak collections are specialized containers of nearly unlimited size that represent a set of data that is in the pipeline. These datasets can be bounded also referred to as fixed size such as the national census data or unbounded such as a Twitter feed or data from weather sensors coming in continuously. P collections are the input and the output of every single transform operation.

Transforms are the data processing steps inside of your pipeline, transforms take one or more of those P collections perform an operation that you specify on each element in that collection and produce one or more P collections as an output. A transform can perform nearly any kind of processing operation including performing mathematical computations on data, converting data from one format to another grouping data together, reading and writing data, filtering data to only the elements that you want or combining data elements into single data values.

Source and sink API's provide functions to read data into and out of collections. The sources act as the roots of the pipeline and the sinks are the endpoints of the pipeline. Cloud dataflow has a set of built-in sinks and sources but it is also possible to write sources and sinks for custom data sources too. Let us look at different pipeline examples to get a sense of the processing capabilities of cloud datflow.

(Refer Slide Time: 04:28)



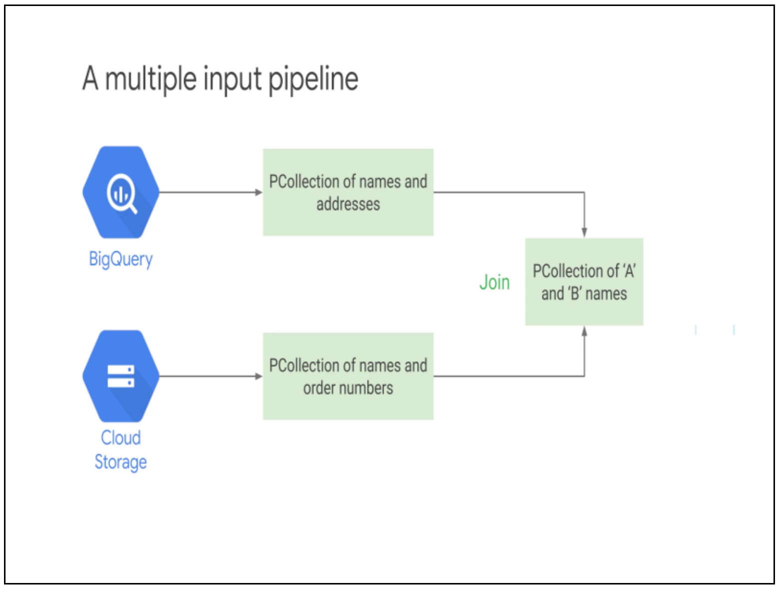
A multiple transform pipeline extract

Transform A: names that start with 'A'
PCollection of 'A' names
Read
PCollection of BigQuery rows
BigQuery
Transform B: names that start with 'B'
PCollection of 'B' names

In this multiple transform pipeline example data read from bigquery is filtered into two collections based on the initial character of the name. Note that the inputs in these examples could be from a different data source and that this pipeline does not go so far as to reflect an output. In this merge pipeline example we are taking the data that was filtered into to collection in our previous multiple transform pipeline an example.

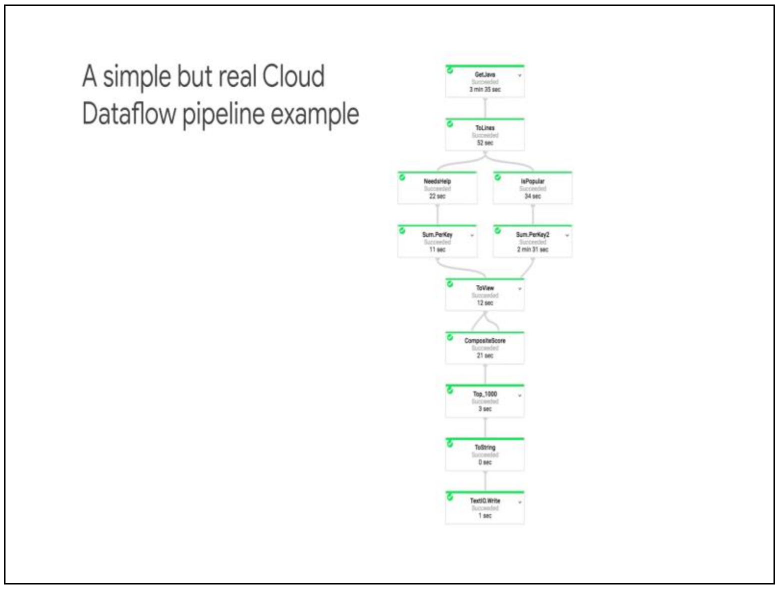
And merging those two datasets together this leaves us with a single data set with names that start with A and B.

(Refer Slide Time: 05:05)


A multiple input pipeline
BigQuery
PCollection of names and addresses
Join
PCollection of 'A' and 'B' names
Cloud Storage
PCollection of names and order numbers

In this multiple input pipeline example we are doing joins from different data sources. The job of cloud dataflow is to ingest data from one or more sources if necessary in parallel transform that data and then load the data into one or more sinks. Google services can be used as both a source and a sink.

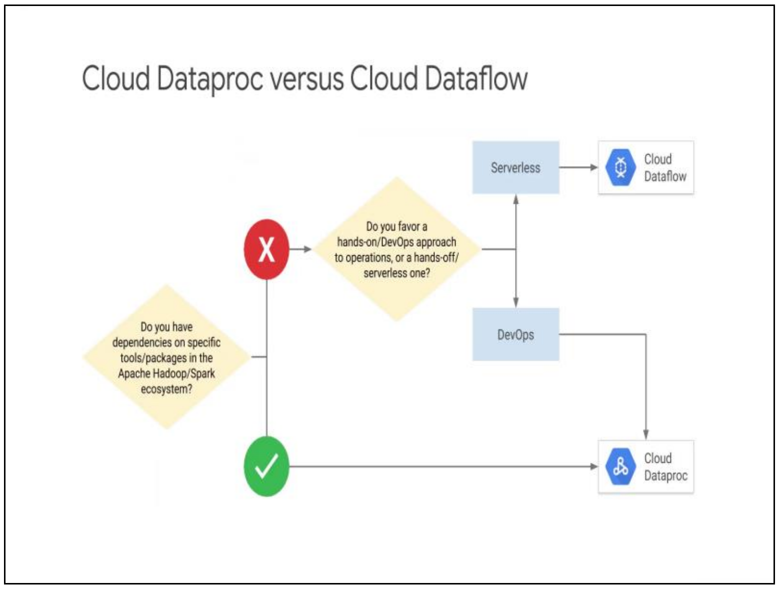
(Refer Slide Time: 05:26)



In this simple but real example the cloud dataflow pipeline reads data from a bigquery table the source processes it in various ways and the transforms writes its output to Google Cloud Storage which is our sink. Some of the transforms in this example our map operations and some are reduce operations. You can build really expressive pipelines. Each step in the pipeline is elastically scaled. So, there is no need to launch and manage your own cluster instead the service provides all the resources on demand.

It has automated and optimized work partitioning built-in which can dynamically rebalance lagging work that reduces the need to worry about hotkeys that is situations would disapprove fortunately large chunks of your input get mapped to the same cluster.

(Refer Slide Time: 06:23)



Cloud Dataproc versus Cloud Dataflow

Do you have dependencies on specific tools/packages in the Apache Hadoop/Spark ecosystem?
X
Do you favor a hands-on/DevOps approach to operations, or a hands-off/ serverless one?
Serverless
Cloud Dataflow
DevOps
Cloud Dataproc

We have discussed cloud dataproc and cloud dataflow as managed service solutions for processing your Big Data. This flow chart summarizes what differentiates one from the other. Both cloud dataproc and cloud dataflow MapReduce operations. The biggest difference between them is that cloud dataproc works similarly as Hadoop would work in the physical infrastructure. You would still create a cluster of servers to perform your ETL jobs.

In the case of cloud dataflow the process is serverless you provide the Java or Python code and leverage the Apache bean SDK to perform ETL operations on batch and streaming data in a serverless fashion.