Illuminating Data Journeys with Apache Beam Lineage

Rohit Sinha

BEAM SUMMIT

September 4-5, 2024

Sunnyvale, CA. USA

Agenda

01 Introduction

02 Motivation

03 Demo

04 Lineage in Beam

05 Beam Lineage On Dataflow

01

Introduction

Data lineage traces the relationship between data sources based on movement of data, explaining how data was sourced and transformed.

Data Lineage Types

Asset Level Lineage

Tracks the relationships between entire datasets or tables

Column Level Lineage

- Tracks the origin and transformations of columns
- Less granular, offering a broader view of data transformations
- Used for impact analysis, schema management

Row Level Lineage

- Tracks the origin and transformation of individual data rows
- More granular, pinpointing the exact source of each row
- Used for data quality, compliance, and debugging purposes

Field Level Lineage

Similar to column-level lineage, but the term "field" is often used in the context of semi-structured or unstructured data formats like JSON or XML, where the structure might not be as rigid as in a traditional relational database

02

Motivation

Customer challenges

01 | **Inability to understand and trust data**

“My manager just asked me if I am using the table from the authoritative source—how can I check this quickly?”

02 | **Inability to do deterministic change management**

“What happens if I drop a table/change a column?”

03 | **Inability to do effective root cause analysis**

“There are issues in the data in a given table—how can I quickly zero in on the potential cause for the issue?”

04 | **Inability to meet compliance requirements effectively**

“How can I guarantee to authorities that I have not used prohibited data in my models to introduce bias?”

05 | **Inability to manage data estate at scale**

“Help me auto curate/auto apply policies based on lineage to automatically manage data”

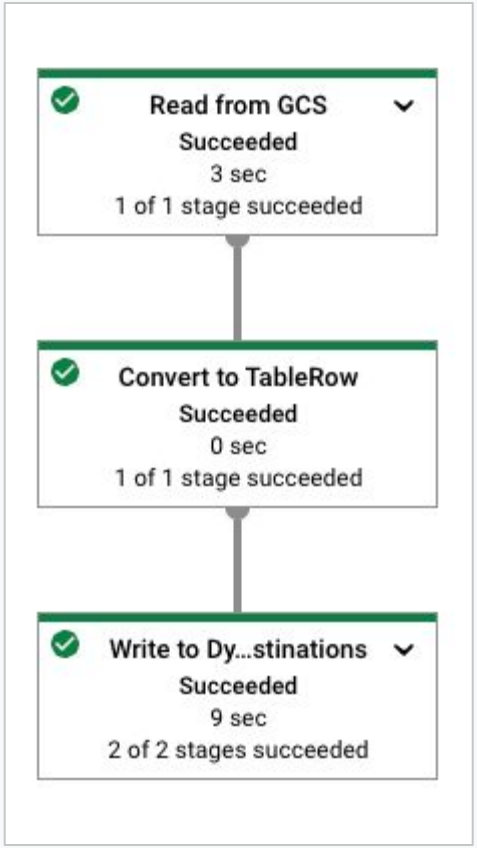
03

Demo

Dynamic GCS To BQ

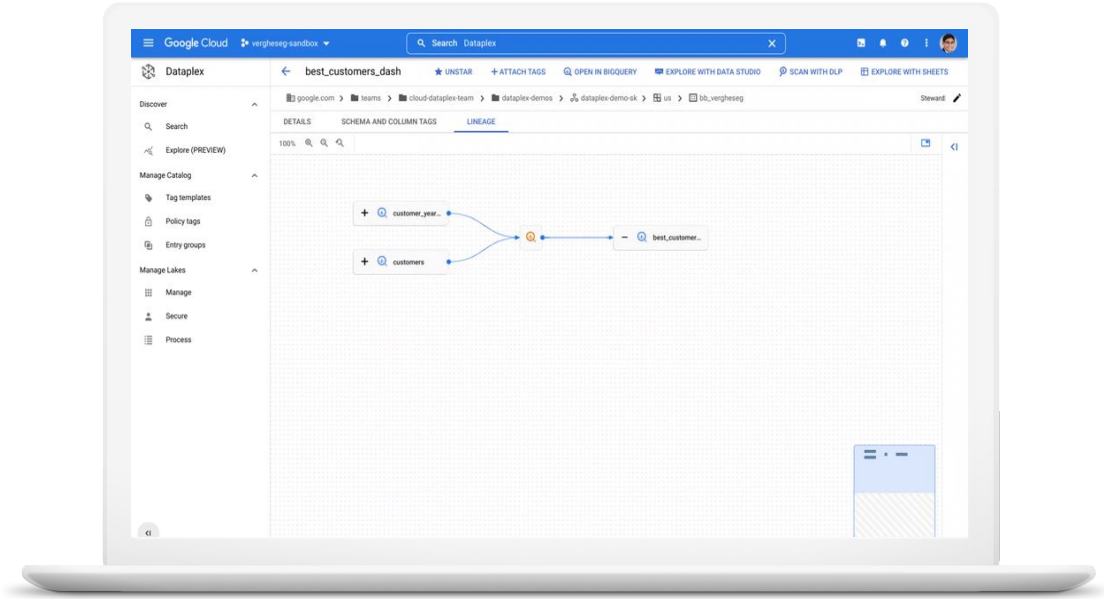
Beam job running on Dataflow

1. Reads from GCS bucket using wildcard pattern in path (i.e. input files are dynamic)
2. The input file path is provided as an argument
3. The file data consists of two columns *type, message*
4. Writes to Bigquery using Dynamic Destination feature wherein the table name is determined by *type* value of the record.



Dataplex Lineage

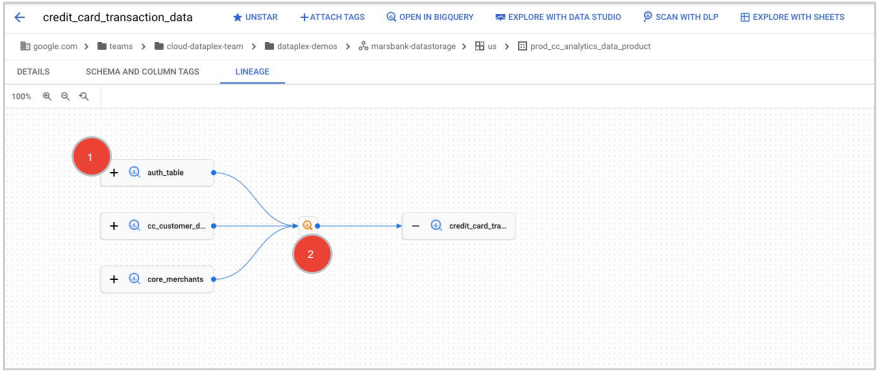
Dataplex lets you track how data moves through your systems: where it comes from, where it is passed to, and what transformations are applied to it.



Concept overview

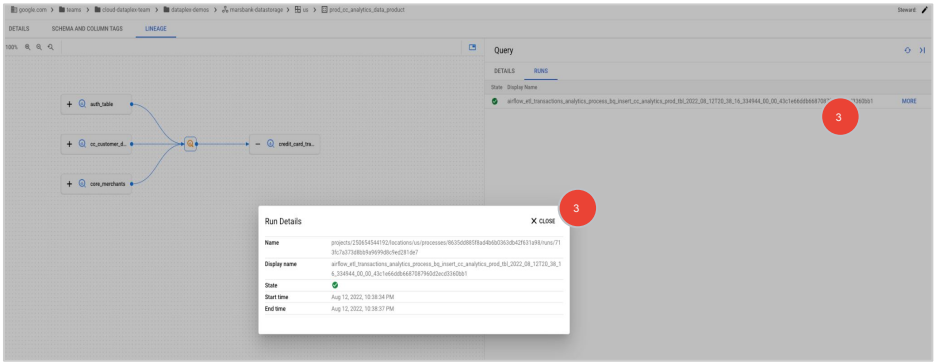
Entry

An entry is an instance of a particular Entry type. It represents the actual Table, Report, Process, defined by the relevant metadata attributes.



Process

A process is the execution template—it is a directive that indicates how an execution should happen in a compute environment. For example, a transformation template in a pipeline or an ad hoc SQL script that is to be executed.



Run

A run is an execution of a process.

Concept overview

Lineage

Lineage is an association between two or more entries or fields in entries based on data movement and models how they are related. Lineage is directed from one or more input entries/fields (say X) to one or more output entries/fields (say Y).

Lineage event

A lineage event is a fact at a point of time reported by another system. It is the granular information using which relationships are generated.

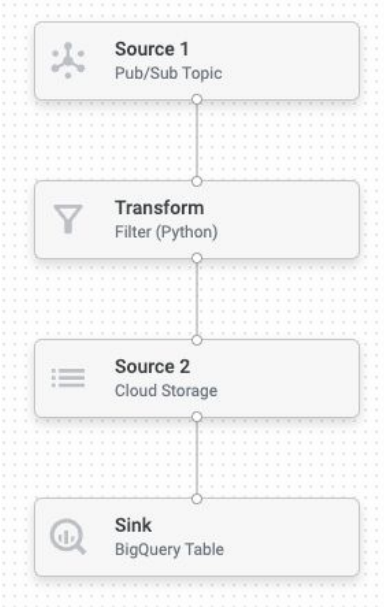
03

Demo

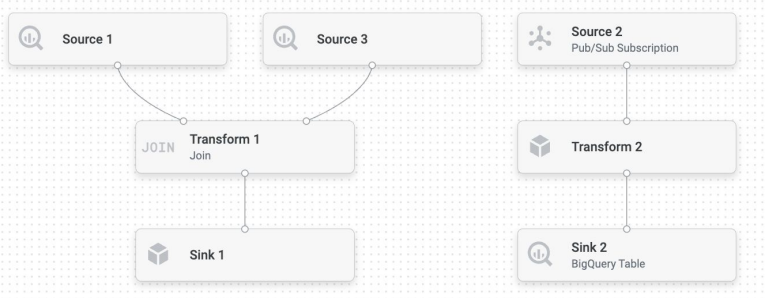
04

Lineage In Beam

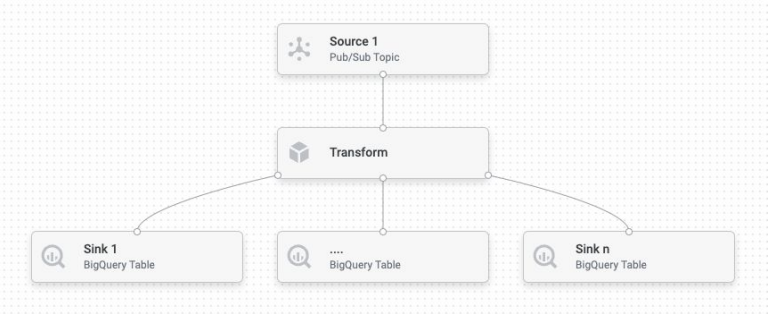
Beam Jobs



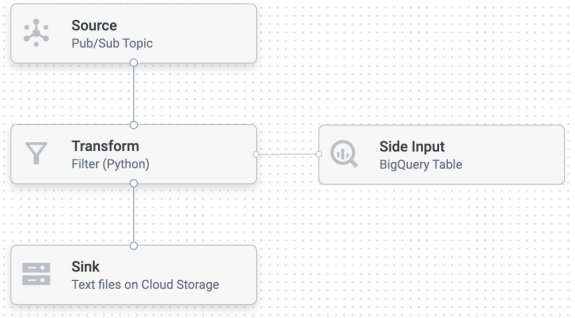
Dynamic Reads



Multiple Sources and Sinks



Dynamic Destinations



Side Input

Metrics in Beam

Visibility

Provide a way to gain visibility into your pipeline's operation as it's running.

Dynamic

Metrics can be created dynamically during runtime, allowing flexibility in instrumenting your pipeline.

Scoped

Each metric update is associated with the specific transform (step) where it was emitted, providing context for the data.

Graceful Degradation

If a runner doesn't fully support metrics, updates may be dropped, but your pipeline won't fail.

Supported Types

- Counter
- Gauge
- Distribution
- Histogram

StringSet as Metric

- Reports set of unique string values
- Allows IOs to report resource locations they are reading/writing.
- Can also be used to report all other sort of things
  - Scheme of the record
  - Type of transform
  - Any JSON structure
- Lineage information via Metrics can easily be provided by *developers* writing IOs
- Can be consumed by *Runners* to build lineage graph.
- Runner don't need to inspect IOs to collect lineage information (brittle approach)

```
public interface StringSet extends Metric {

 void add(String value);

 default void add(String... values;

}
```

Lineage Class

- Wraps StringSet Metrics
- Provides a namespace *lineage* and named metrics *sources* and *sinks* for reporting source/sink information.

```
public class Lineage {

 public static final String LINEAGE_NAMESPACE =
"lineage";
 private static final Lineage SOURCES = new
Lineage(Type.SOURCE);
 private static final Lineage SINKS = new
Lineage(Type.SINK);

 private final StringSet metric;

 public static Lineage getSources();

 public static Lineage getSinks();

}
```

Emitting Lineage

- Easy
- Can be emitted at Runtime just like other metrics
- Can be emitted from any IO including custom IOs

GcsFileSystem

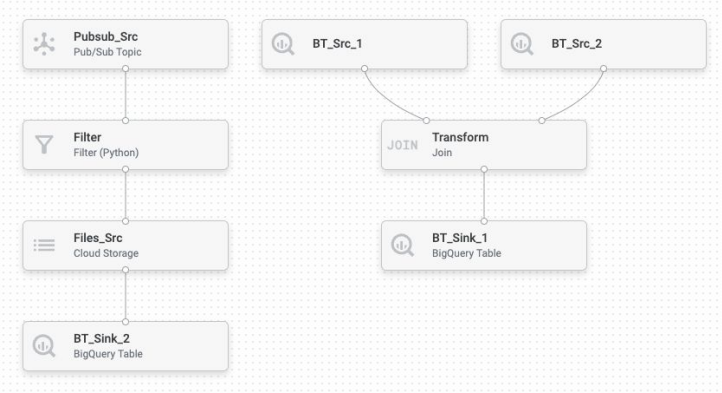
```
Lineage.getSources().add(gcsFilepath);
```

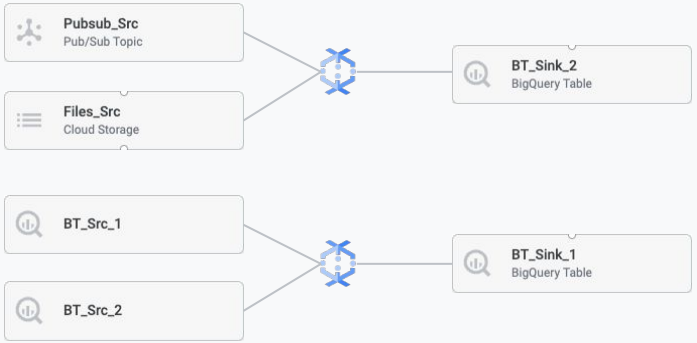
BigTable

```
Lineage.getSinks().add(bigTableID);
```

Lineage Graph

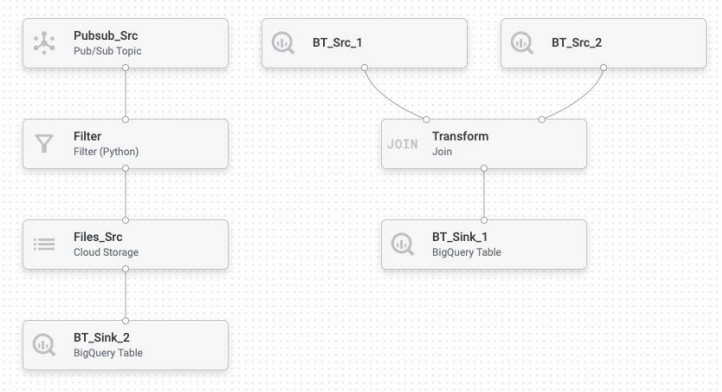
- The lineage information from IOs and Beam job graph is sufficient to build a lineage graph.
- Traverse the **job graph** for each sink and find all sources in path
- Special consideration: Dynamic Reads, Composite nodes
- Single step can report multiple datasets
- Lineage graph can change through the lifecycle of job





Supporting Column Level Lineage

- It is also possible to extend this architecture to build Column Level Lineage in Beam
- Individual Transform can emit:
  - Input schema
  - Operation performed
  - Output Schema
- Possibly in well established Open Lineage CLL format.
- OpenLineage is an open-source project focused on lineage collection. It standardizes the capture and access of lineage about data pipelines and datasets.
- Like before the CLL information can then be combined with job graph to build CLL for the Beam job.



Lineage Support In Beam

Supported SDKs

- Java
- Python
- Go

Supported IOs

- Pubsub
- BigQuery
- GCS
- Cloud Bigtable
- Kafka
- and more...

Lineage on Runners

- Add StringSet metric support
- Build lineage graph links
- Report to lineage visualization tools
- Reach out to us for support

Future Plans

- Support for emitting CLL in Transforms
- Out of box CLL support in Beam YAML

05

Beam Lineage On Dataflow

Dataflow Lineage

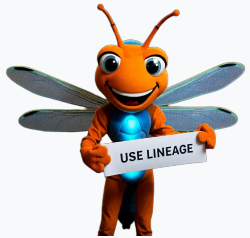
Dataflow

- Upcoming asset level lineage launch
- Lineage metrics are pushed to Dataflow service backend
- Dataflow service creates lineage graphs which are then published to GCP Dataplex for visualization.

To use

- Update to latest Beam SDK
- Launch Beam job on Dataflow with `--dataflowServiceOptions=enable_lineage`

Asset Level Lineage Support Matrix	
Batch Jobs	✔
Streaming Jobs	✔
Templates	✔
Data Pipelines	✔
Runner V2	✔



Thank you!

Questions?

linkedin.com/in/rohitsinha54
rosinha@google.com

BEAM SUMMIT