Introducing Managed IO: the New Era of Beam Connectors

Ahmed Abualsaud

Agenda

How “normal” pipelines look

Updating to a new SDK

Transform-level upgrading with Managed IOs

How does transform upgrade work exactly?

Runner modifying transform configuration

Dataflow runner demo

How “normal” Beam pipelines usually look

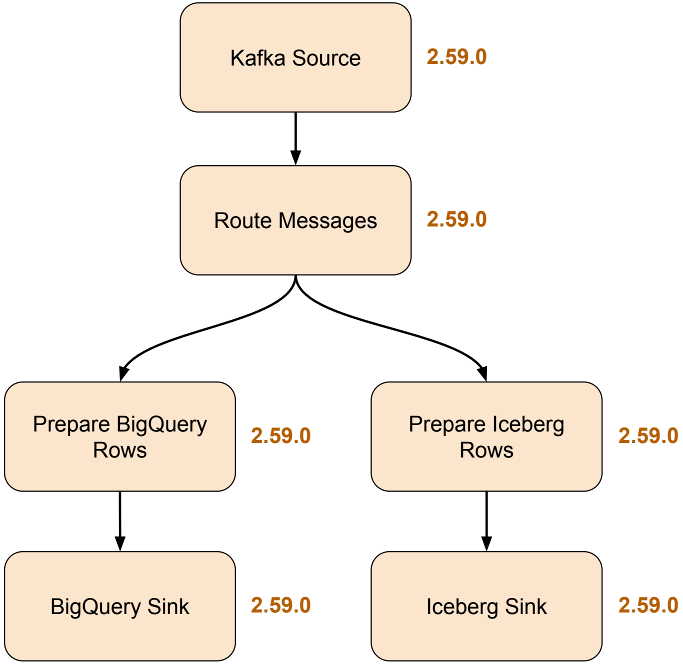
```
implementation "org.apache.beam:beam-sdks-java-core:2.59.0 "
implementation "org.apache.beam:beam-sdks-java-io-iceberg:2.59.0 "
implementation "org.apache.beam:beam-sdks-java-io-kafka:2.59.0 "
implementation "org.apache.beam:beam-sdks-java-io-google-cloud-platform:2.59.0 "
```

```
Pipeline p = Pipeline.create(options);
PCollectionTuple split = p
        .apply(KafkaIO.readBytes()
                .withTopic(topic)
                .withBootstrapServers(address))
        .apply(RouteMessages.of());

split.get("to_bigquery")
        .apply(PrepareBigQueryRows.of())
        .apply(BigQueryIO.write()
                .to(tableSpec)
                .withMethod(STORAGE_WRITE_API)
                .withTriggeringFrequency(Duration.standardSeconds(5))
                .withSchema(tableSchema));

split.get("to_iceberg")
        .apply(PrepareIcebergRows.of())
        .apply(IcebergIO.writeRows(catalogConfig)
                .to(tableIdentifier)
                .withTriggeringFrequency(Duration.standardSeconds(5)));
```

Pipeline SDK version = 2.59.0



Updating to a newer SDK

Pros

Particular IOs have

- New features
- Bug fixes
- Performance improvements

Taking care of security vulnerabilities

Cons

Maintenance overhead

Compatibility with the rest of your project

Going through dependency hell

Potential regression in other parts of the SDK

Enter Transform Service and Managed IOs

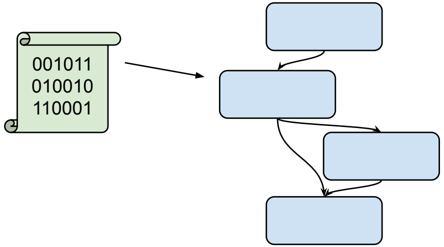
Service that replaces individual transforms

Transform identifier

```
--transformsToOverride=beam:schematransform:org.apache.beam:iceberg_write:v1
--transformServiceBeamVersion=2.65.0
```

How does this work?

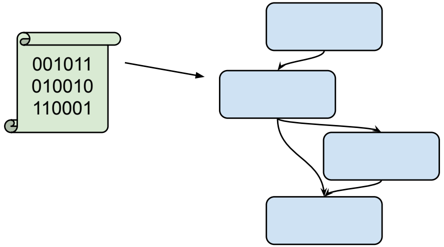
Beam **translates** the transform's configuration into bytes and stores it in the proto pipeline graph.



The service can extract this configuration and deserialize to recreate the transform using the same configuration, but in a different version.

Enter Transform Service and Managed IOs

Beam **translates** the transform's configuration into bytes and stores it in the proto pipeline graph.



The service can extract this configuration and deserialize to recreate the transform using the same configuration, but in a different version.

Managed IOs

Are IOs that implement the required translation logic (bytes <-> configuration) to be eligible for replacement.

Unified interface:

Java

```
Managed.write(ICEBERG)
    .withConfig(Map.of(
        “table”, “abc.xyz”,
        “triggering_frequency_seconds”, 10,
        “catalog_name”, “my_catalog”,
        ...))
```

Python

```
beam.managed.Write(“iceberg”, config={
        “table”, “abc.xyz”,
        “triggering_frequency_seconds”, 10,
        “catalog_name”, “my_catalog”,
        ...})
```

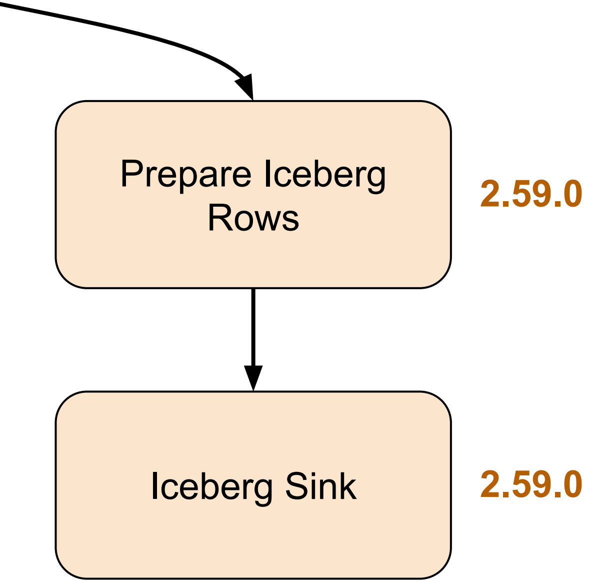
https://beam.apache.org/documentation/io/managed-io/

Managed IOs

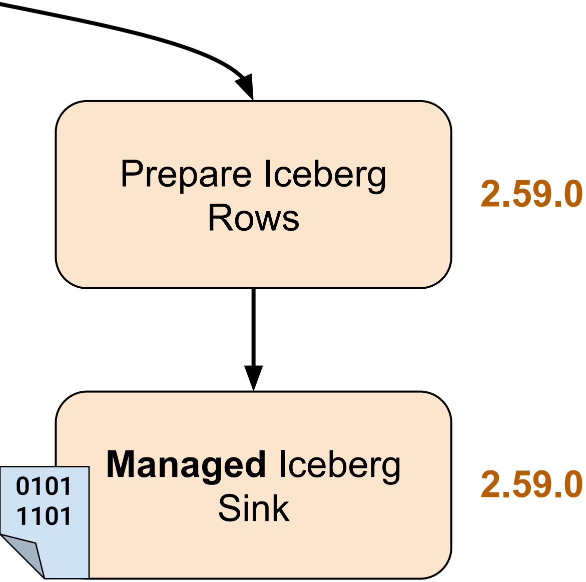
SDK and Runner have the ability to:

- **Upgrade the transform**
- Modify the transform’s configuration

How does this work?



How does this work?



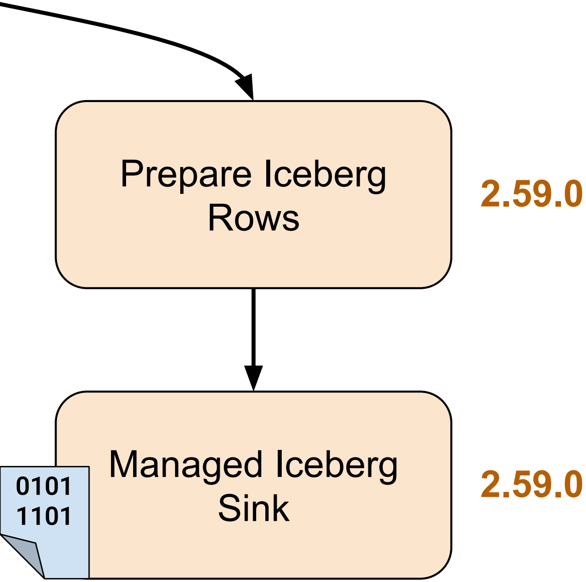
Prepare Iceberg Rows

2.59.0

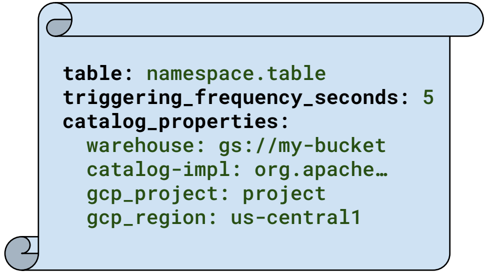
Managed Iceberg Sink

2.59.0

How does this work?

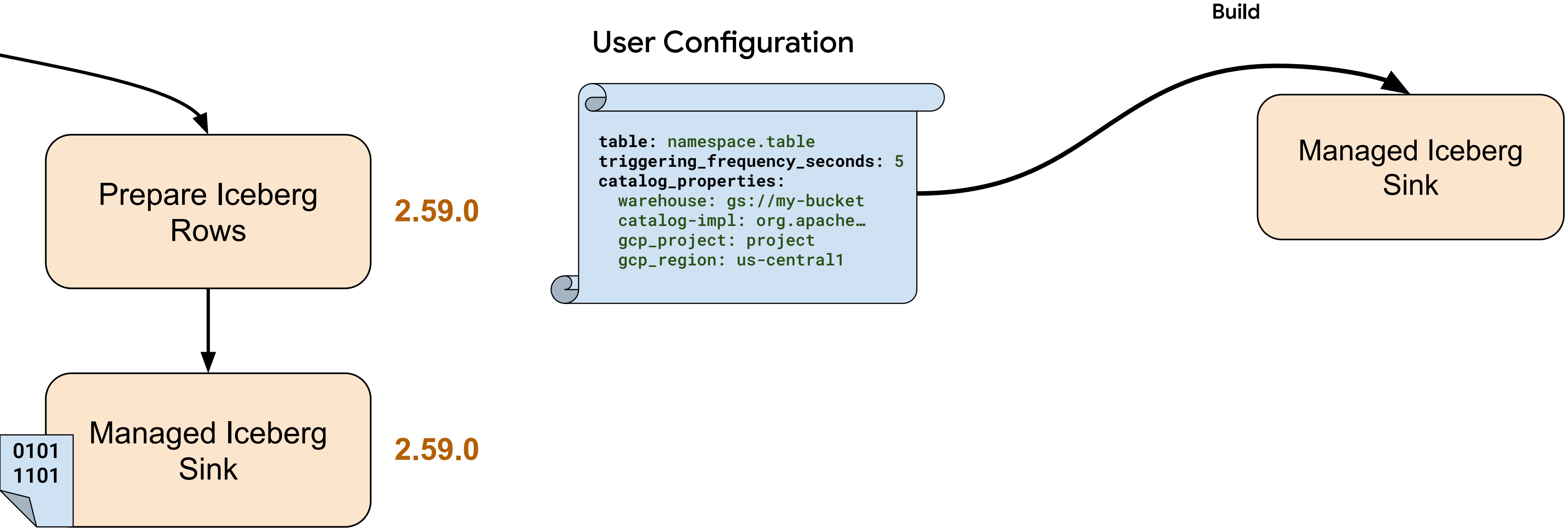


User Configuration



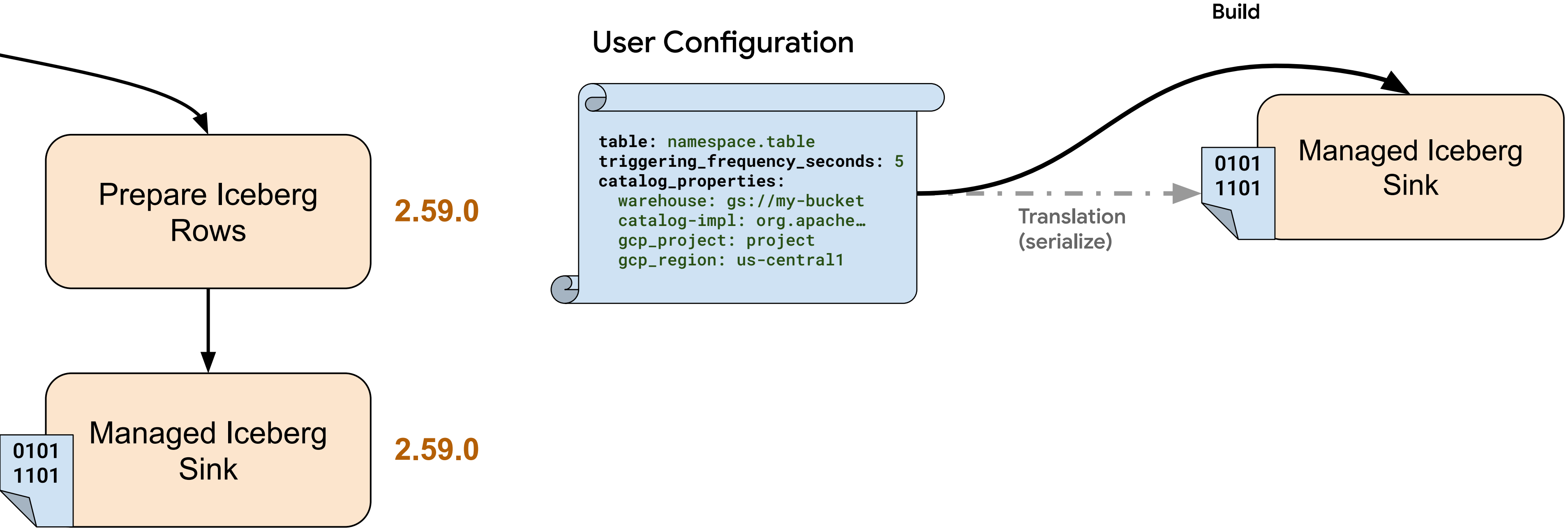
```
table: namespace.table
triggering_frequency_seconds: 5
catalog_properties:
  warehouse: gs://my-bucket
  catalog-impl: org.apache…
  gcp_project: project
  gcp_region: us-central1
```

How does this work?



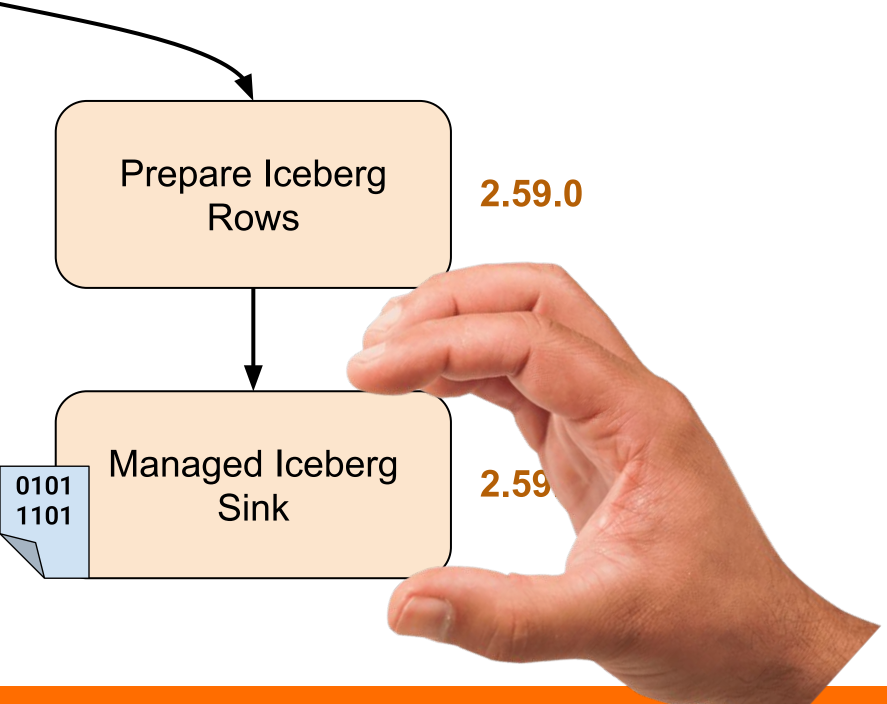
```
table: namespace.table
triggering_frequency_seconds: 5
catalog_properties:
  warehouse: gs://my-bucket
  catalog-impl: org.apache…
  gcp_project: project
  gcp_region: us-central1
```

How does this work?



```
table: namespace.table
triggering_frequency_seconds: 5
catalog_properties:
  warehouse: gs://my-bucket
  catalog-impl: org.apache…
  gcp_project: project
  gcp_region: us-central1
```

How does this work?



Prepare Iceberg Rows

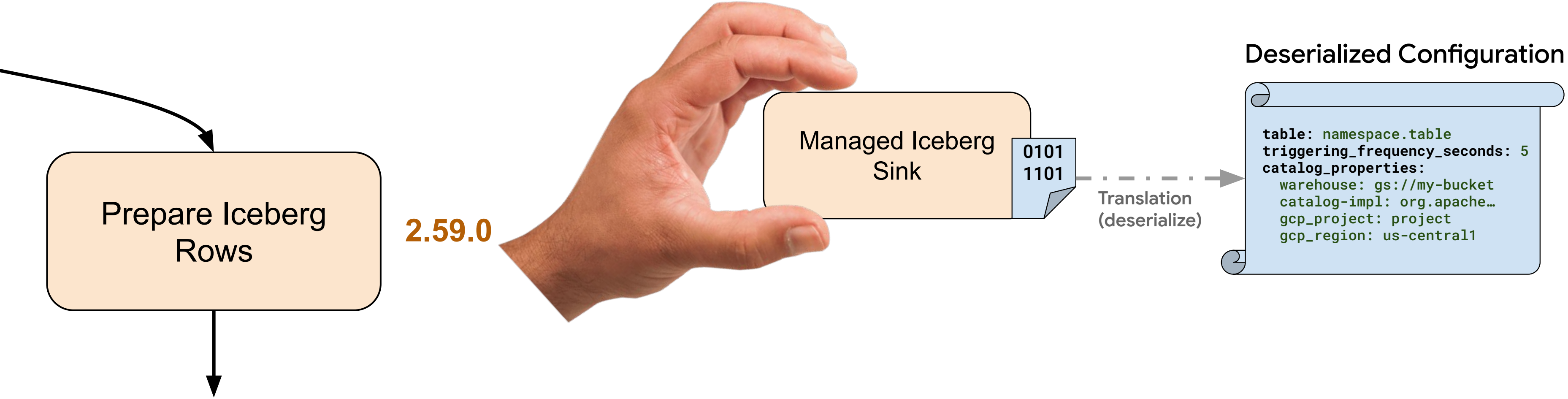
2.59.0

Managed Iceberg Sink

0101
1101

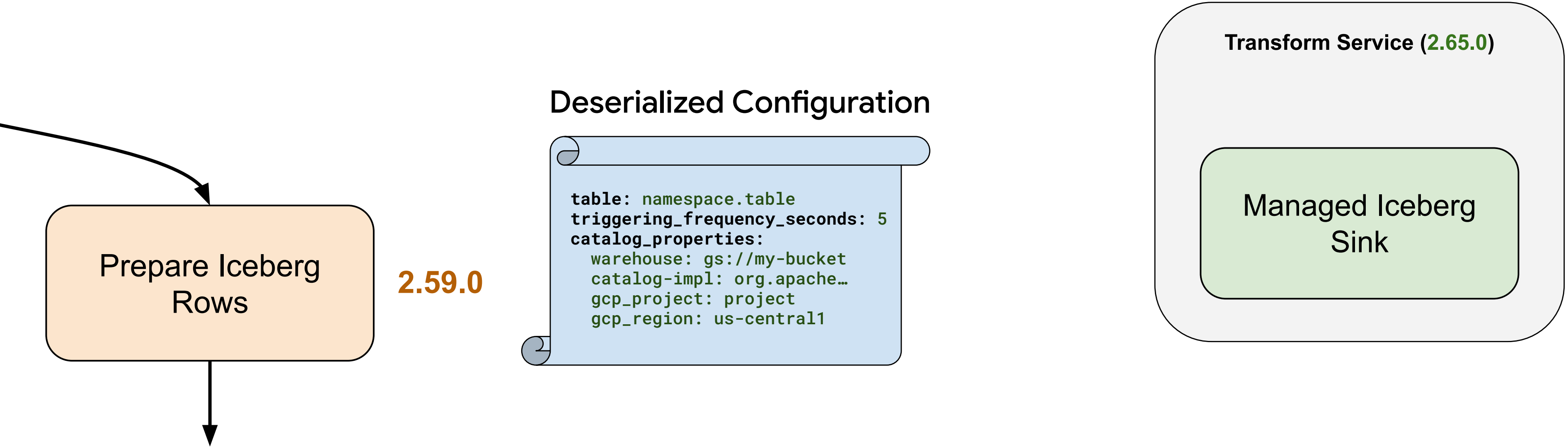
2.59

How does this work?



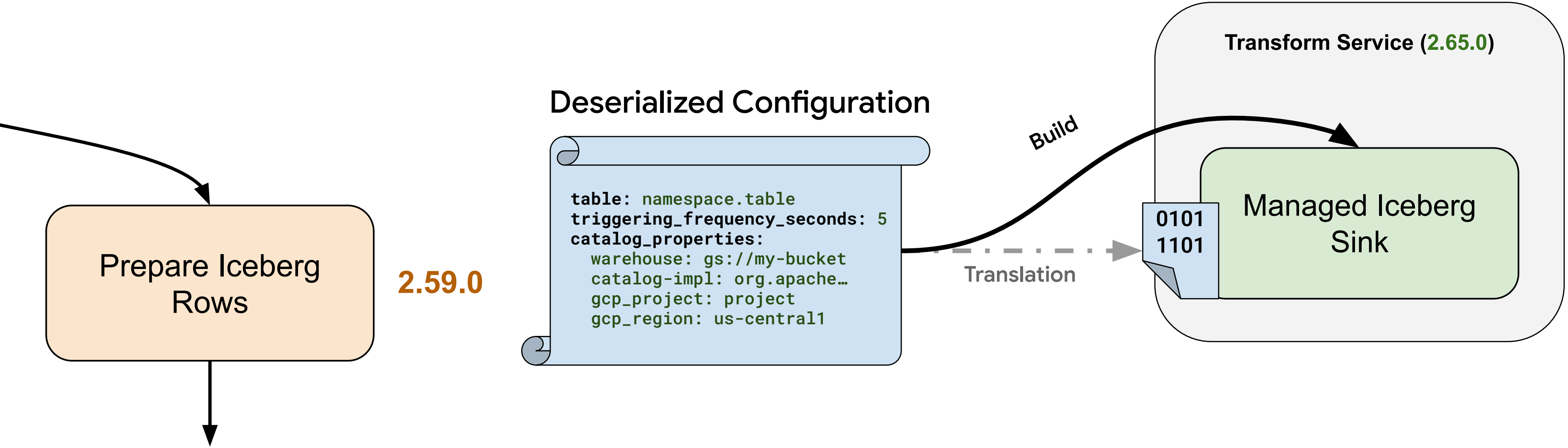
```
table: namespace.table
triggering_frequency_seconds: 5
catalog_properties:
  warehouse: gs://my-bucket
  catalog-impl: org.apache…
  gcp_project: project
  gcp_region: us-central1
```

How does this work?



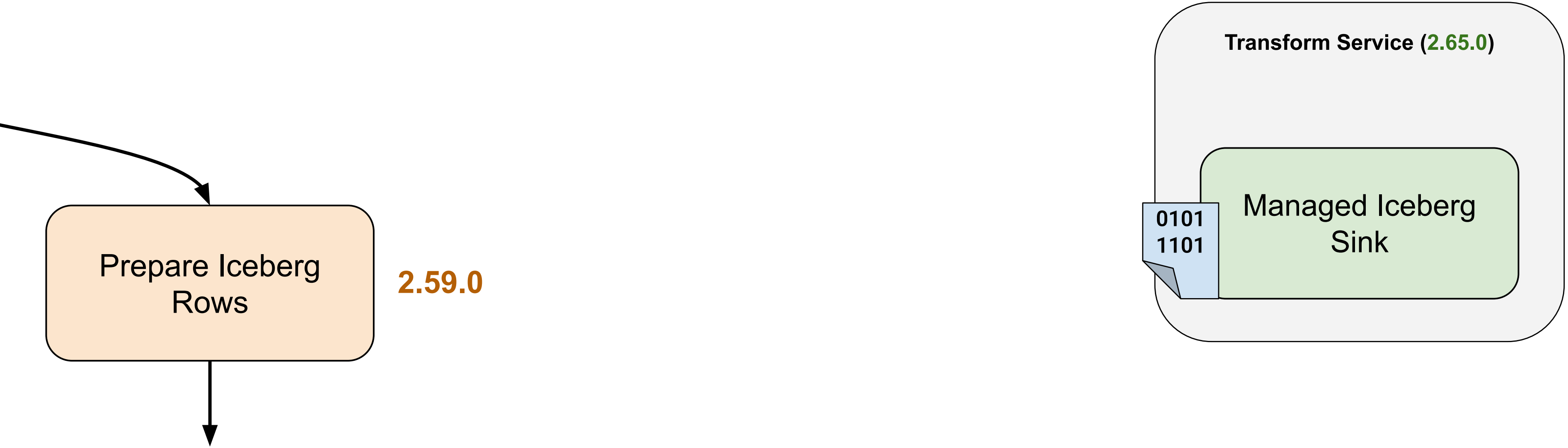
```
table: namespace.table
triggering_frequency_seconds: 5
catalog_properties:
  warehouse: gs://my-bucket
  catalog-impl: org.apache…
  gcp_project: project
  gcp_region: us-central1
```

How does this work?

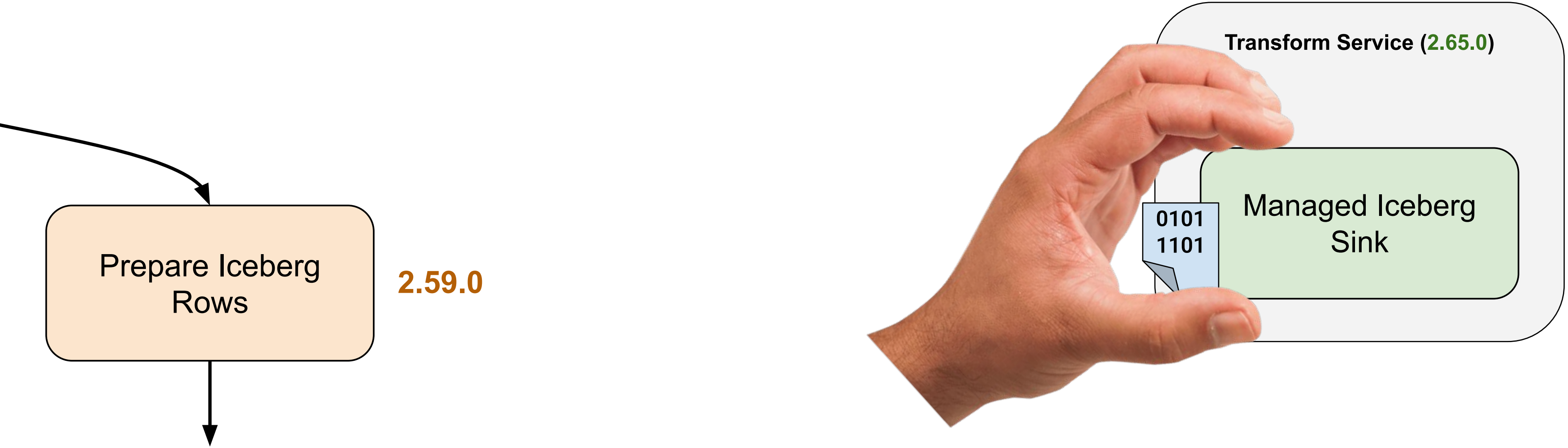


```
table: namespace.table
triggering_frequency_seconds: 5
catalog_properties:
  warehouse: gs://my-bucket
  catalog-impl: org.apache…
  gcp_project: project
  gcp_region: us-central1
```

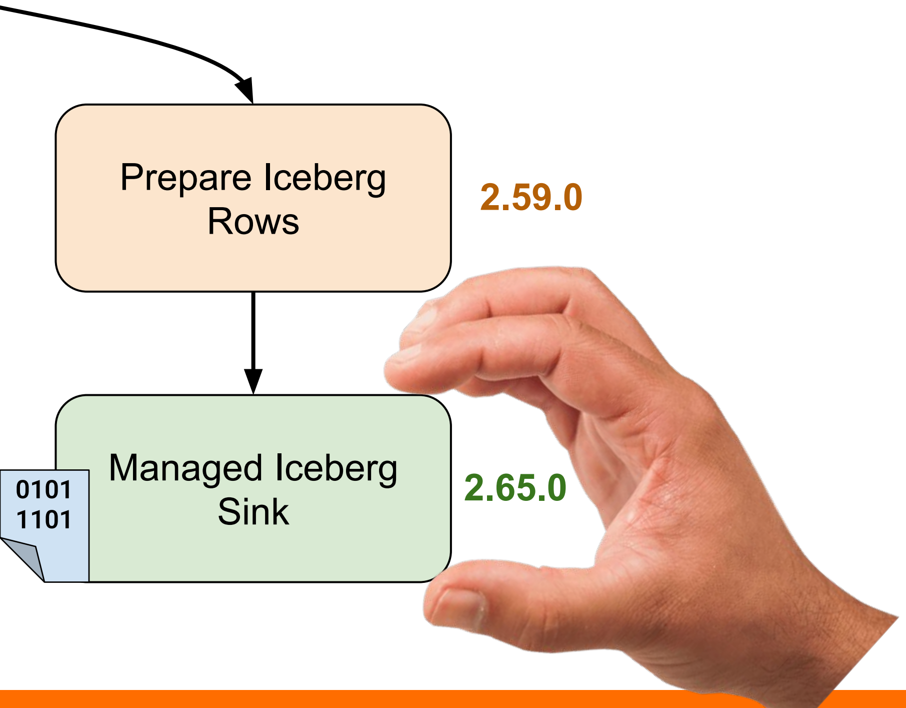
How does this work?



How does this work?



How does this work?



Prepare Iceberg Rows

2.59.0

Managed Iceberg Sink

2.65.0

0101
1101

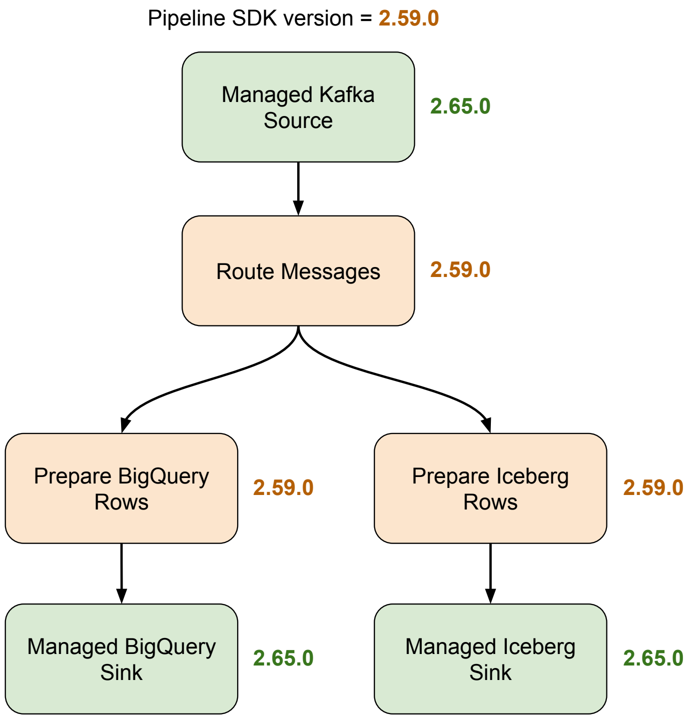
Beam pipeline with Managed IOs

```
implementation "org.apache.beam:beam-sdks-java-core:2.59.0 "
implementation "org.apache.beam:beam-sdks-java-managed:2.59.0 "
implementation "org.apache.beam:beam-sdks-java-io-iceberg:2.59.0 "
implementation "org.apache.beam:beam-sdks-java-io-kafka:2.59.0 "
implementation "org.apache.beam:beam-sdks-java-io-google-cloud-platform:2.59.0 "
```

```
PCollectionTuple split = p
        .apply(Managed.read(KAFKA).withConfig(
                Map.of("format", "RAW",
                       "topic", "test-topic",
                       "bootstrap_servers", "host:port")))
        .apply(ParseMessages.of());

split.get("to_bigquery")
        .apply(PrepareBigQueryRows.of())
        .appl(Managed.write(BIGQUERY).withConfig(
                Map.of("table", tableSpec)));

split.get("to_iceberg")
        .apply(PrepareIcebergRows.of())
        .apply(Managed.write(ICEBERG).withConfig(
                Map.of("table", tableIdentifier,
                       "catalog_properties", catalogProps,
                       "triggering_frequency_seconds", 5)));
```



Demo (Portable runner)

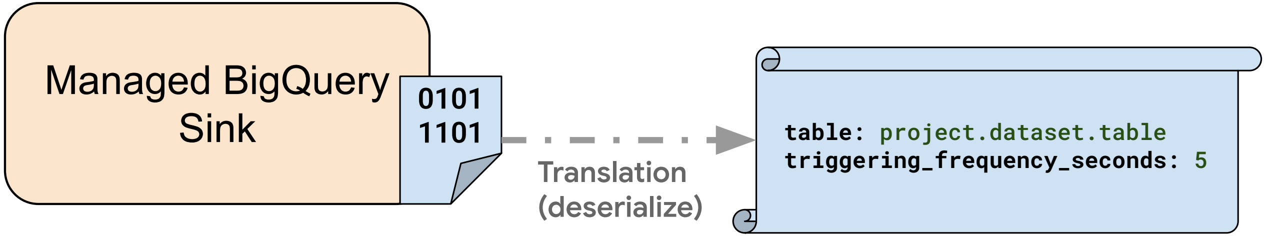
Managed IOs

Are IOs that implement the required translation logic (bytes <-> configuration) to be eligible for replacement.

SDK and Runner can use this information to:

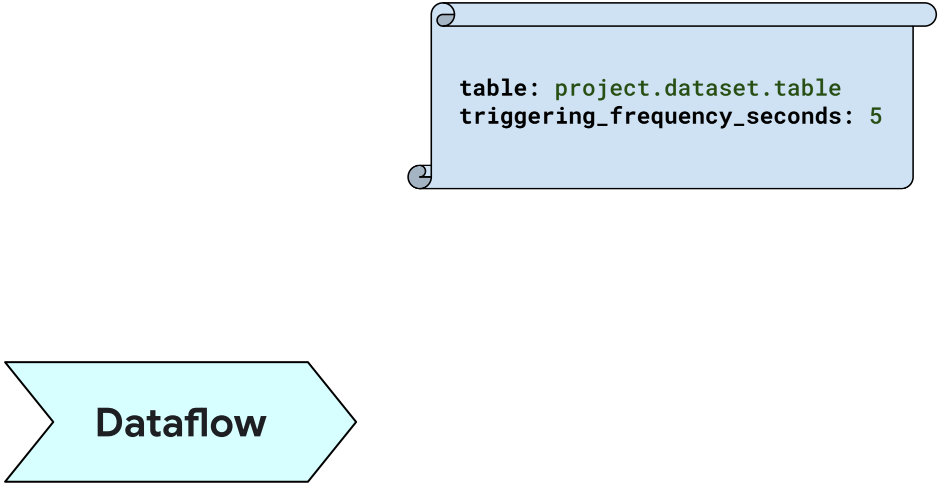
- Upgrade the transform
- **Modify the transform’s configuration**

Modifying the transform config



```
table: project.dataset.table
triggering_frequency_seconds: 5
```

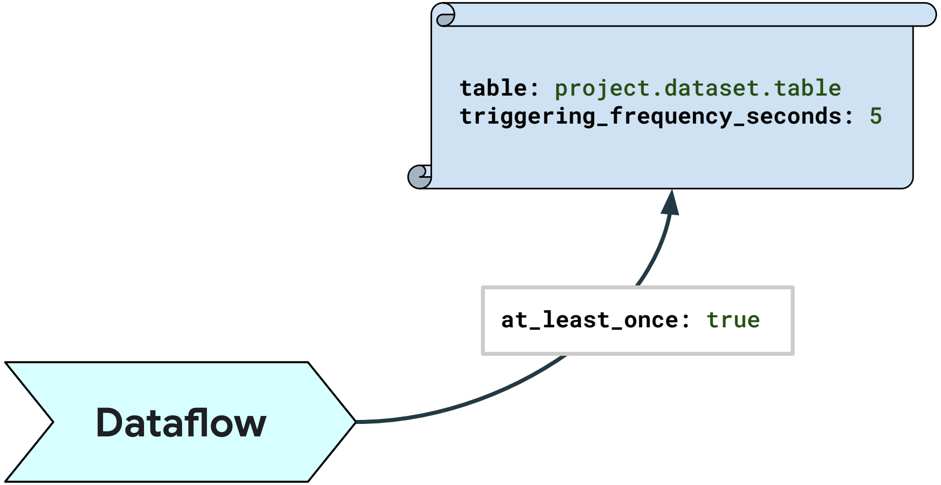
Modifying the transform config



```yaml
table: project.dataset.table
triggering_frequency_seconds: 5
```

```
--dataflowServiceOptions=streaming_mode_at_least_once
```

Modifying the transform config

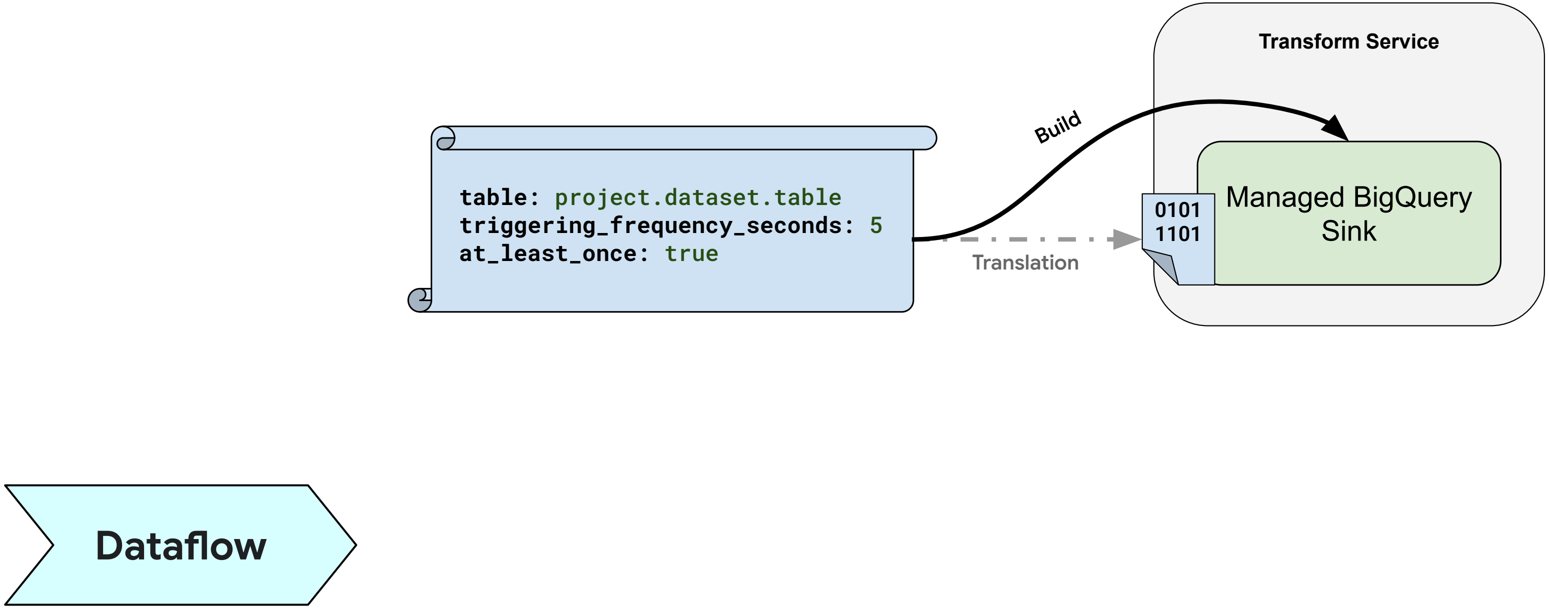


```
table: project.dataset.table
triggering_frequency_seconds: 5
```

```
at_least_once: true
```

```
--dataflowServiceOptions=streaming_mode_at_least_once
```

Modifying the transform config



```
table: project.dataset.table
triggering_frequency_seconds: 5
at_least_once: true
```

```
--dataflowServiceOptions=streaming_mode_at_least_once
```

Dataflow Runner V2

- Automatically detects Managed IOs and upgrade them to the latest version.
- When updating a streaming pipeline, it will first attempt to upgrade to a newer version.

Demo (Dataflow runner)

Thank you!

Contact:

github.com/ahmedabu98

linkedin.com/in/ahmedabu98

a.abualsaud98@gmail.com