

# 常に使えるデータを！ BigQuery でデータ パイプラインを構築

饗庭 秀一郎

Google Cloud,

Data Analytics Specialist

# Table of Contents

データ パイプラインとは	01
--------------	----

Dataform	02
----------	----

01

# データパイプラインとは

# データパイプライン

発生したデータをビジネスに使えるデータに継続的に処理するための仕組み



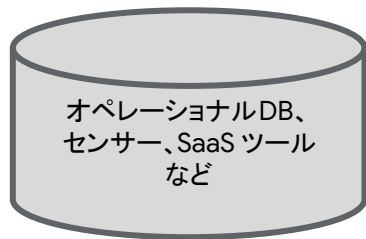
- オペレーショナルデータベース
- センサ
- モバイルログ

- BI
- ML
- 他システム
- マーケティングツール連携

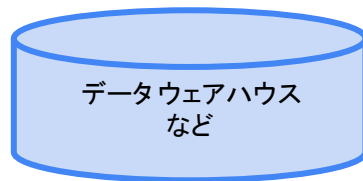
# 分析に使えるデータとは

発生した生データと分析に使えるデータの間にあるギャップ

## 発生した生データ



## 分析に使えるデータ



BigQuery

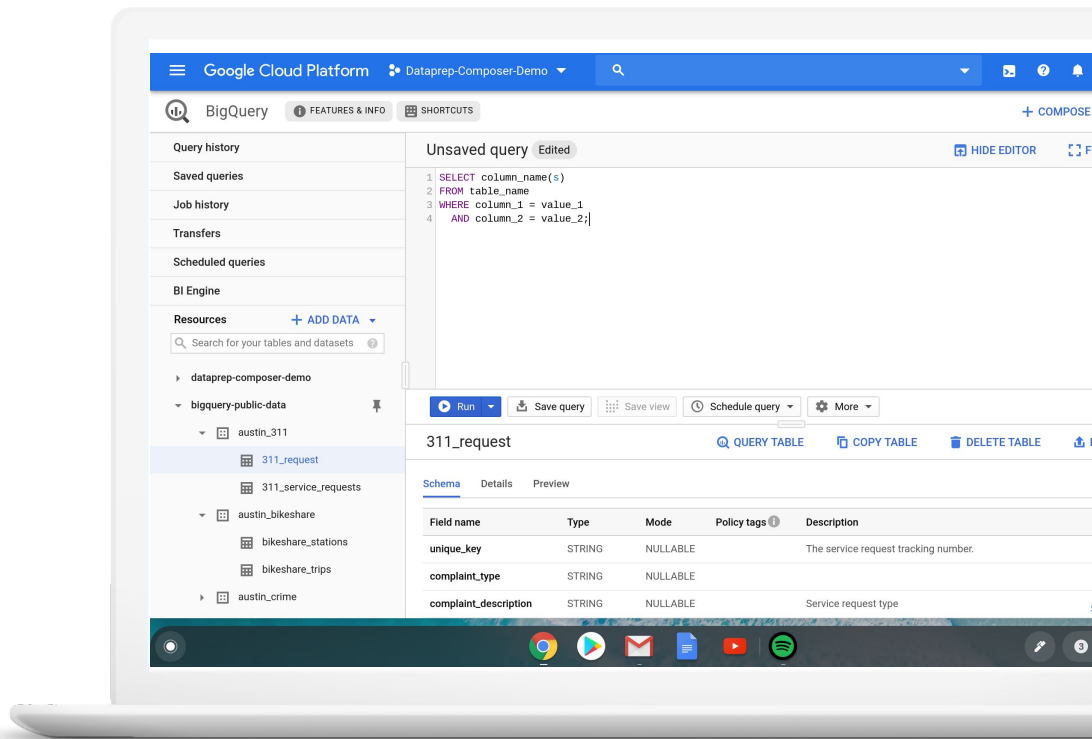
- データの重複・欠損・ノイズ
- 分析したいデータのサイロ化
- ビジネス ロジック
- ダウンストリームに適したデータ形式
- 半構造化・非構造化データ
- 個人情報マスキング などなど

# BigQuery

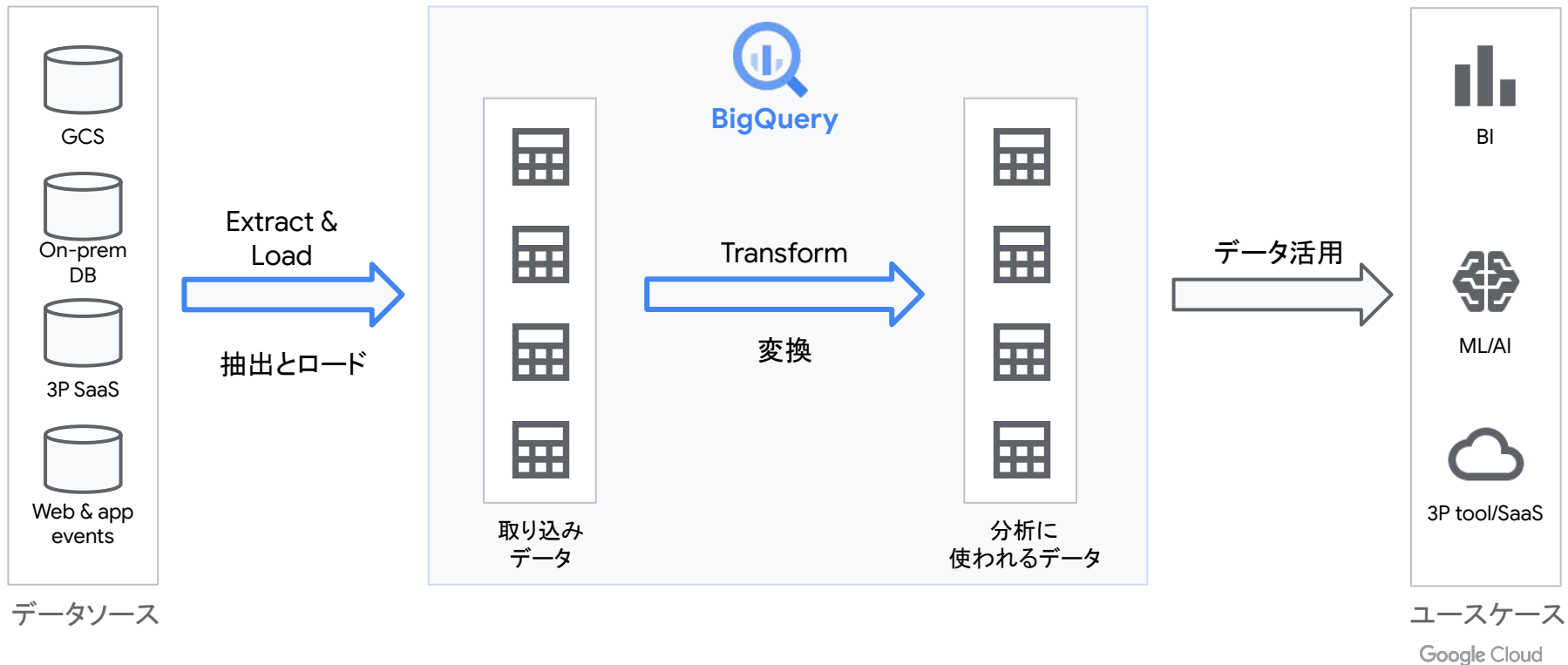
Google Cloud の  
エンタープライズ向け  
レイクハウス

ギガバイトからエクサバイト級のストレージや SQL クエリ処理

フルマネージドで  
サーバレス、高い可用性や  
セキュリティを担保



# BigQuery を中心としたデータ パイプライン



# データパイプラインに求められる要件

データパイプラインには主に「ワークフロー」と「データ処理」の2つの機能が求められる

## データ処理

E: Extract(抽出)、L: Load(ロード)、T: Transform(変換)の基本的処理を大規模なデータに対してもスケーラブルに実行



## ワークフロー

複数の処理の実行とその依存関係・順序を制御し、リアルタイムやバッチのような実行タイミングも管理





# BigQuery へのデータ取り込み

様々なデータの種別や取り込み要件に対応



Cloud Storage

ログデータなどの  
一括取り込み



Datastream

RDBMS の変更  
データ差分取り込み



Pub/Sub

ストリーミング データの  
リアルタイム取り込み



BigQuery  
ネイティブ ストレージ

**どうやって取り込んだデータを  
分析できるデータにする？**


# BigQuery での変換の実装方法

## BigQuery の従来機能

- クエリのスケジュール
  - ビュー
  - マテリアライズド・ビュー
- 

## Dataform

---

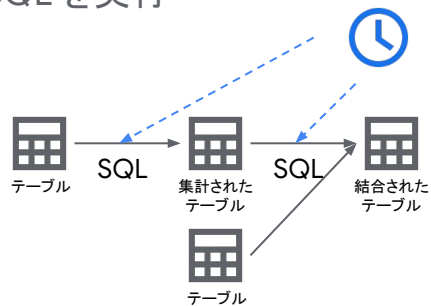
- 
- 処理がシンプル
  - 規模が小さい

- 処理が複雑
- 規模が大きい

# BigQuery の機能による変換のシンプルな実装

## クエリのスケジュール

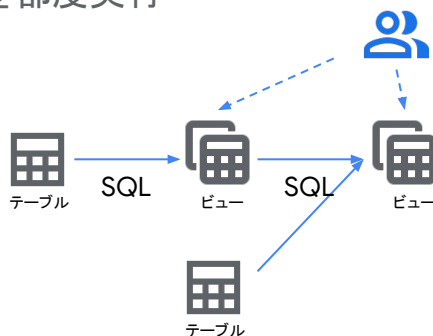
指定した時間間隔で  
SQL を実行



- ✓ すべてのクエリをサポート
- ✓ 複数ステートメントクエリで簡単なフローを実現可能
- ✗ リアルタイムなデータ反映が困難
- ✗ 依存関係が管理できない

## ビュー

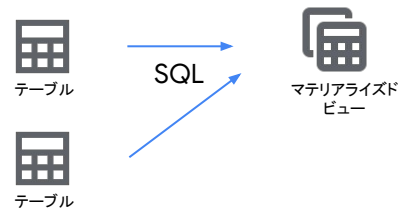
ビュー参照時に定義した SQL  
を都度実行



- ✓ すべてのクエリをサポート
- ✓ リアルタイムな反映が可能
- ✗ 多段の場合パフォーマンスやコストの観点から非効率的

## マテリアライズド・ビュー

元テーブルの変更を自動的に  
反映して常に最新化



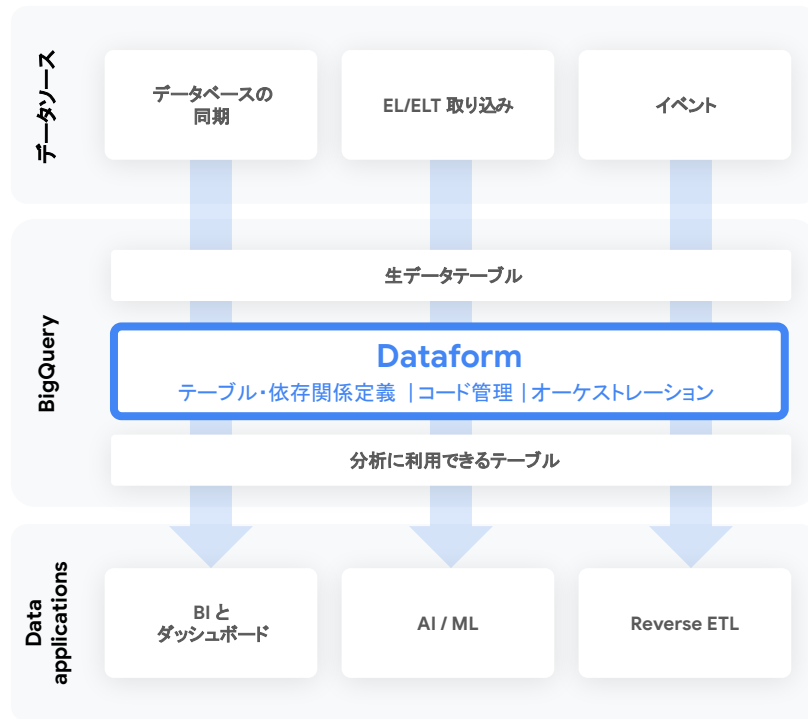
- ✓ リアルタイムな反映が可能
- ✗ クエリのサポートが部分的(一部の結合や関数などが非サポート)
- ✗ 多段構成が組めない

02

# Dataform

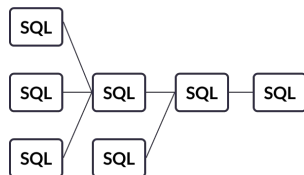
# Dataform Preview

- BigQuery の SQL をデータ処理に利用した Transform 実装のためのフルマネージドサービス
- テーブルの定義やテーブルの依存関係を定義型の SQLX ファイルに記述
- SQL が扱えるスキルがあればデータアナリストなどもセルフでパイプラインを構築可能
- 実装はコードかつそれらをコードレポジトリで管理するためソフトウェアの開発プロセスやチーム開発を導入可能



# Dataform のアーキテクチャ

```
1 config {
2   type: "table",
3   assertions: {
4     uniqueKey: ["station_id"],
5   }
6 }
7
8 SELECT
9   start_station_id AS station_id,
10  start_station_name AS name,
11  COUNT(*) AS trips
12 FROM
13   `bigquery-public-data.new_york_citibike.citibike_trips`
14 GROUP BY station_id, name
```



## SQLX の開発

BigQuery に統合された UI 上でテーブル定義の SQLX ファイルを作成し、GitHub などと連携

## SQL コンパイルとワークフロー

SQLX に記述されたテーブル間の依存関係を解析し、ワークフローとして構築

## BigQuery でクエリ実行

スケジューラや外部ツールをトリガーやワークフローに沿ってクエリを実行

## テーブル生成

テーブル・ドキュメント・データ品質のテスト結果などを生成

# Dataform の画面

BigQuery のコンソールに統合

## Web IDE

The screenshot shows the Dataform Web IDE interface. The top bar displays the project name 'nyc-citibike', a 'START EXECUTION' button, and a 'GO TO EXECUTIONS' button. A 'COMPILED' status indicator is visible in the top right. The left sidebar shows a file explorer with a tree view containing files like 'partition\_test.sql', 'station\_master.sql', and 'station\_trips.sql'. The main editor area shows the content of 'definitions/station\_trips.sql', which includes a configuration block and a SQL query. The right sidebar shows the 'METADATA' for the 'station\_trips' table, including its full name, type, and dependencies.

```
1 config {
2   type: "table",
3   assertions: {
4     uniqueKey: ["station_id"],
5   }
6 }
7
8 SELECT
9   start_station_id AS station_id,
10  start_station_name AS name,
11  COUNT(1) AS trips
12 FROM
13   `bigquery-public-data.new_york_citibike.citibike_trips`
14 GROUP BY station_id, name
```

**station\_trips**

Full name: random-322508.dataform.station\_trips

Type: Table

**dataform\_station\_trips\_ass...**

Full name: random-322508.dataform\_assertions.dataform\_station\_trips\_assertions\_unique\_key\_0

Type: Assertion

Dependencies: dataform.station\_trips

## ワークフローの実行結果

The screenshot shows the 'WORKFLOW EXECUTION LOGS' page in the Dataform Web IDE. The page title is 'demo' and it includes buttons for 'CREATE DEVELOPMENT WORKSPACE' and 'CONNECT WITH GIT'. The main content area displays a table of execution logs. The table has columns for Status, Start time, Duration, Source, and Contents. The logs show a series of successful executions (green checkmarks) and one failed execution (red exclamation mark).

DEVELOPMENT WORKSPACES WORKFLOW EXECUTION LOGS

This page shows all the past Dataform executions in your repository

Filter: Enter property name or value

Status	Start time	Duration	Source	Contents
✓	<a href="#">Sep 30, 2022, 2:05:28 PM</a>	6 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 30, 2022, 2:02:33 PM</a>	7 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 30, 2022, 1:56:32 PM</a>	5 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 30, 2022, 1:54:29 PM</a>	47 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 30, 2022, 1:42:17 PM</a>	5 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 30, 2022, 1:41:17 PM</a>	5 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 30, 2022, 1:39:27 PM</a>	7 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 29, 2022, 5:06:51 PM</a>	5 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 29, 2022, 5:06:15 PM</a>	5 seconds	nyc-citibike	1 action
!	<a href="#">Sep 29, 2022, 5:03:38 PM</a>	3 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 29, 2022, 5:02:21 PM</a>	7 seconds	nyc-citibike	1 action
✓	<a href="#">Sep 29, 2022, 5:01:46 PM</a>	5 seconds	nyc-citibike	1 action



# SQLX

生成されるテーブルやビューごとに作成する SQLX ファイル

テーブル設定

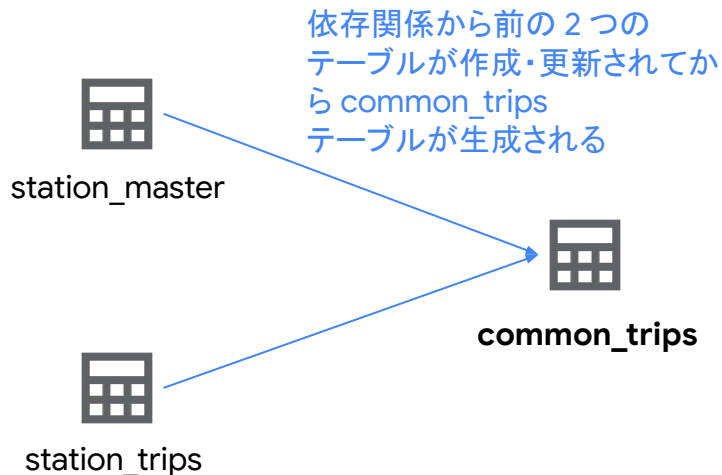
カラム説明

データ品質テスト

テーブルと依存関係の  
定義

```
config {  
  type: "table",  
  description: "The station master table w/ the # of trips",  
  columns: {  
    station_id: "Unique station ID",  
    name: "Station name",  
  },  
  assertions: {  
    uniqueKey: ["station_id", "name"],  
    nonNull: ["name"]  
  }  
}  
  
SELECT  
  station_id, name,  
  latitude, longitude, region_id,  
FROM ${ref('station_master')} stations  
LEFT OUTER JOIN ${ref('station_trips')} trips  
USING (station_id, name)
```

# テーブルの依存関係の実装



common\_trips.sqlx

```
SELECT
  station_id AS station_id,
  name AS name,
  latitude,
  longitude,
  region_id,
FROM `${ref('station_master')}` stations
LEFT OUTER JOIN `${ref('station_trips')}` trips
USING (station_id, name)
```

## ref 関数

Dataform の中心となるビルトイン関数でテーブル間の依存関係を記述  
`\${ref(<sqlxファイル名>)}`とすることで他のテーブルを参照

# SQLX ドキュメント

```
config {  
  type: "table",  
  description: "The station master table w/ the # of trips",  
  columns: {  
    station_id: "Unique station ID",  
    name: "Station name",  
  }  
}
```

## BigQuery のテーブル メタデータ

### Table info

Table ID	random-322508.dataform.trips_common
Table size	134.37 KB
Long-term storage size	0 B
Number of rows	1,584
Created	Sep 12, 2022, 11:05:49 PM UTC+9
Last modified	Sep 12, 2022, 11:05:49 PM UTC+9
Table expiration	NEVER
Data location	US
Default collation	
Description	The station master table w/ the # of trips

GitHub などコードレポジトリで SQLX ファイルを管理できるため、このようなデータ アクセシビリティのための情報もガバナンスを効かせて管理可能

Field name	Type	Mode	Collation	Default Value	Policy Tags	Description
<input type="checkbox"/> <a href="#">station_id</a>	INTEGER	NULLABLE				Unique station ID
<input type="checkbox"/> <a href="#">name</a>	STRING	NULLABLE				Station name
<input type="checkbox"/> <a href="#">latitude</a>	FLOAT	NULLABLE				
<input type="checkbox"/> <a href="#">longitude</a>	FLOAT	NULLABLE				
<input type="checkbox"/> <a href="#">region_id</a>	INTEGER	NULLABLE				
<input type="checkbox"/> <a href="#">rental_methods</a>	STRING	NULLABLE				
<input type="checkbox"/> <a href="#">CAPACITY</a>	INTEGER	NULLABLE				
<input type="checkbox"/> <a href="#">num_bikes_available</a>	INTEGER	NULLABLE				

# Dataform のアサーション

```
assertions: {
  uniqueKey: ["station_id", "name"],
  nonNull: ["region_id"],
}
```

region\_id に NULL がある  
レコードが存在する例

実行結果でエラーがでる

Details

Start time: Sep 13, 2022, 12:21:46 AM

Status: ❌ Failed

Duration: 13 seconds

Source: nyc-citibike

Actions

Filter: Enter property name or value

Status	Start time ↑	Duration	Action
✅	Sep 13, 2022, 12:21:46 AM	3 seconds	station_master
✅	Sep 13, 2022, 12:21:46 AM	6 seconds	station_trips
✅	Sep 13, 2022, 12:21:50 AM	2 seconds	dataform_station_master_assertions_rowCo...
✅	Sep 13, 2022, 12:21:50 AM	2 seconds	dataform_station_master_assertions_unique...
✅	Sep 13, 2022, 12:21:53 AM	3 seconds	trips_common
❌	Sep 13, 2022, 12:21:57 AM	2 seconds	dataform_trips_common_assertions_rowCon...
✅	Sep 13, 2022, 12:21:57 AM	2 seconds	dataform_trips_common_assertions_uniqueK...

## assertion でチェックできるものの例

- nonNull - NULL 値を許容しない
- rowConditions - カラムに対する任意の条件
- uniqueKey - カラムの値の一意性

Assertion 結果出力ビューで実際に違反しているレコードが見れる

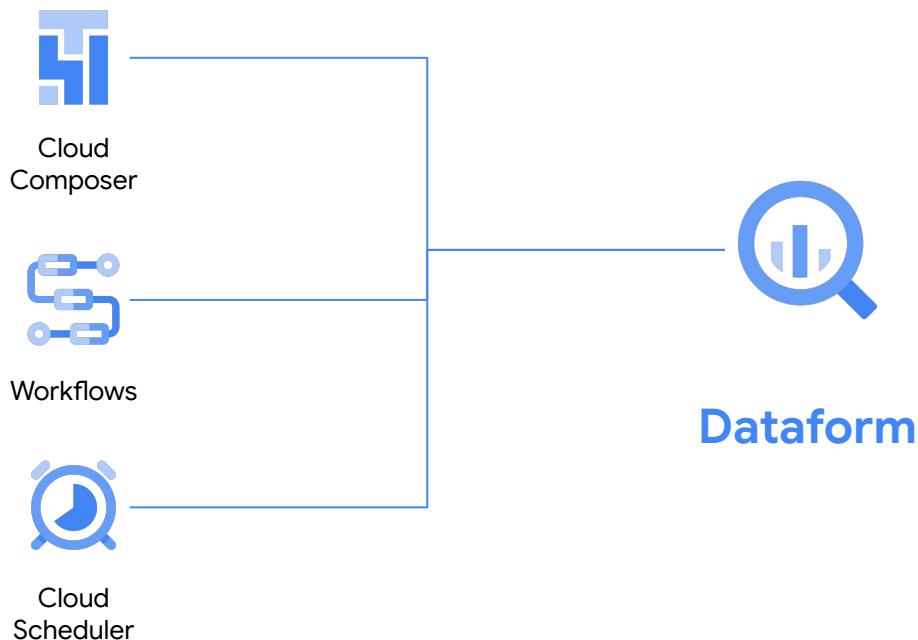
Query results

1 SELECT  
2 \*  
3 FROM  
4 `random-322508.dataform\_assertions.dataform\_trips\_common\_assertions\_rowConditions`  
5 LIMIT  
6 1000

Row	failing_row_condition	station_id	name	latitude	lon...
1	region_id IS NOT NULL	4683	Broadway & W 61 St	40.7700301...	-7...
2	region_id IS NOT NULL	4671	Calyer St & Guernsey St	40.7275580...	-7...
3	region_id IS NOT NULL	4664	Prospect Ave & Greenwood Ave	40.651745	
4	region_id IS NOT NULL	4656	Clinton St & Newark St	40.73743	
5	region_id IS NOT NULL	4644	51 St & Hobart St	40.7576723...	-7...
6	region_id IS NOT NULL	4674	45 Ave & 21 St	40.7473707...	-7...

# 実行制御

現在 Dataform はスケジューリング機能など実行契機を管理する機能がないため、外部のサービスから制御する



# Dataform のユースケース



データ エンジニア

主要なデータパイプラインの Transform の部分に使い、データ品質を担保



データアナリスト/  
データサイエンティスト

主要パイプラインで作られたすでに BigQuery にあるデータを個々の分析要件にあったデータに変換、あるいは ML の前処理の実装



**Thank you.**