Assessing Data Quality with Dataplex

Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

1. Click **Activate Cloud Shell** at the top of the Google Cloud console.

2. Click through the following windows:

   - Continue through the Cloud Shell information window.
   - Authorize Cloud Shell to use your credentials to make Google Cloud API calls.

When you are connected, you are already authenticated, and the project is set to your **Project_ID**, `PROJECT_ID`. The output contains a line that declares the **Project_ID** for this session:

```
Your Cloud Platform project in this session is set to "PROJECT_ID"
```

`gcloud` is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

3. (Optional) You can list the active account name with this command:

```
gcloud auth list
```

4. Click **Authorize**.

Output:

```
ACTIVE: *
ACCOUNT: "ACCOUNT"

To set the active account, run:
    $ gcloud config set account `ACCOUNT`
```

5. (Optional) You can list the project ID with this command:

```
gcloud config list project
```

Output:

```
[core]
project = "PROJECT_ID"
```

Note: For full documentation of `gcloud`, in Google Cloud, refer to the gcloud CLI overview guide.

Enable Dataproc API

1. In the Google Cloud Console, enter **Cloud Dataproc API** in the top search bar.
2. Click on the result for **Cloud Dataproc API** under Marketplace.
3. Click **Enable**.

Task 1. Create a lake, zone, and asset in Dataplex

To define and run data quality tasks, you first need to create some Dataplex resources.

In this task, you create a new Dataplex lake to store ecommerce customer information, add a raw zone to the lake, and then attach a pre-created BigQuery dataset as a new asset in the zone.

Create a lake

1. In the Google Cloud Console, in the **Navigation menu** (☰), navigate to **Analytics** > **Dataplex**.

If prompted `Welcome to the new Dataplex experience`, click **Close**.

2. Under **Manage lakes**, click **Manage**.
3. Click **Create lake**.

4. Enter the required information to create a new lake:

Property	Value
Display Name	`Ecommerce Lake`
ID	Leave the default value.
Region	_____

Leave the other default values.

5. Click **Create**.

It can take up to 3 minutes for the lake to be created.

Add a zone to the lake

1. On the **Manage** tab, click on the name of your lake.
2. Click **+ADD ZONE**.
3. Enter the required information to create a new zone:

Property	Value
Display Name	`Customer Contact Raw Zone`
ID	Leave the default value.
Type	**Raw zone**

Data locations	**Regional**

Leave the other default values.

For example, the option for **Enable metadata discovery** under **Discovery settings** is enabled by default and allows authorized users to discover the data in the zone.

4. Click **Create**.

It can take up to 2 minutes for the zone to be created.

Note: You can perform the next task once the status of the zone is **Active**.

Attach an asset to a zone

1. On the **Zones** tab, click on the name of your zone.
2. On the **Assets** tab, click **+ADD ASSET**.
3. Click **Add an asset**.
4. Enter the required information to attach a new asset:

Property	Value
Type	**BigQuery dataset**
Display Name	`Contact Info`
ID	Leave the default value.
Dataset	_____.customers

Leave the other default values.

5. Click **Done**.

6. Click **Continue**.

7. For **Discovery settings**, select **Inherit** to inherit the Discovery settings from the zone level, and then click **Continue**.

8. Click **Submit**.

Task 2. Query a BigQuery table to review data quality

In the previous task, you created a new Dataplex asset from a BigQuery dataset named **customers** that has been pre-created for this lab. This dataset contains a table named **contact_info** which contains raw contact information for customers of a fictional ecommerce company.

In this task, you query this table to start identifying some potential data quality issues that you can include as checks in a data quality job. You also identify another precreated dataset that you can use to store data quality job results in a later task.

1. In the Google Cloud Console, in the **Navigation menu** (☰), navigate to **BigQuery**.

2. In the Explorer pane, expand the arrow next to your project ID to list the contents: ____

In addition to the **customer_contact_raw_zone** dataset created by Dataplex to manage that zone, there are two BigQuery datasets that were precreated for this lab:

- customers
- customers_dq_dataset

The dataset named **customers** contains one table named **contact_info**, which contains contact information for customers such as a customer ID, name, email, and more. This is the table that you explore and check for data quality issues throughout this lab.

The dataset named **customers_dq_dataset** does not contain any tables. When you define a data quality job in a later task, you use this dataset as the destination for a new table containing the data quality job results.

3. In the SQL Editor, click on **+ SQL query**. Paste the following query, and then click **Run**:

```
SELECT * FROM `.customers.contact_info`
ORDER BY id
LIMIT 50
```

Copied!

content_copy

This query selects 50 records from the original table and orders the records by the customer id in the results.

4. Scroll through the results in the **Results** pane.

Notice that some records are missing customer IDs or have incorrect emails, which can make it difficult to manage customer orders.

Task 3. Create and upload a data quality specification file

Dataplex data quality check requirements are defined using CloudDQ YAML specification files. Once created, the YAML specification file is uploaded to a Cloud Storage bucket that is made accessible to the data quality job.

The YAML file has four keys sections:

- a list of rules to run (either pre-defined or customized rules)
- row filters to select a subset of data for validation
- rule bindings to apply the defined rules to the table(s)
- optional rule dimensions to specify the types of the rules that the YAML file can contain

In this task, you define a new YAML specification file for data quality checks that identify null customer IDs and emails in the specified BigQuery table. After you define the file, you upload it to a pre-created Cloud Storage bucket for use in a later task to run the data quality job.

Create the data quality specification file

1. In Cloud Shell, run the following command to create a new empty file for the data quality specification:

```
nano dq-customer-raw-data.yaml
```

2. Paste the following code:

```
metadata_registry_defaults:
  dataplex:
    projects: Project ID
    locations: Region
    lakes: ecommerce-lake
    zones: customer-contact-raw-zone
row_filters:
  NONE:
    filter_sql_expr: |-
      True
  INTERNATIONAL_ITEMS:
    filter_sql_expr: |-
      REGEXP_CONTAINS(item_id, 'INTNL')
rule_dimensions:
  - consistency
  - correctness
  - duplication
  - completeness
  - conformance
  - integrity
  - timeliness
  - accuracy
rules:
  NOT_NULL:
    rule_type: NOT_NULL
    dimension: completeness
  VALID_EMAIL:
    rule_type: REGEX
    dimension: conformance
    params:
      pattern: |-
        ^[^@]+[@]{1}[^@]+$
rule_bindings:
  VALID_CUSTOMER:
    entity_uri: bigquery://projects/Project
ID/datasets/customers/tables/contact_info
    column_id: id
    row_filter_id: NONE
    rule_ids:
      - NOT_NULL
  VALID_EMAIL_ID:
    entity_uri: bigquery://projects/Project
ID/datasets/customers/tables/contact_info
    column_id: email
    row_filter_id: NONE
    rule_ids:
      - VALID_EMAIL
```

3. Review the code to identify the two primary data quality rules that are defined in this file.

The `dq-customer-raw-data.yaml` file begins with key parameters to identify the Dataplex resources including the project ID, region, and names of the Dataplex lake and zone.

Next, it specifies the allowed rule dimensions and two primary rules:

- The rule for **NOT_NULL** values refers to the completeness dimension such as null values.
- The rule for **VALID_EMAIL** values refers to the conformance dimension such as invalid values.

Last, the rules are bound to entities (tables) and columns using rule bindings for data quality validation:

- The first rule binding named **VALID_CUSTOMER** binds the **NOT_NULL** rule to the **id** column of the **contact_info** table, which will validate if the ID column has any NULL values.
- The second rule binding named **VALID_EMAIL_ID** binds the **VALID_EMAIL** rule to the **email** column of the **contact_info** table, which will check for valid emails.

4. Enter `Ctrl+X`, then `Y`, to save and close the file.

Upload the file to Cloud Storage

- In Cloud Shell, run the following command to upload the file to a Cloud Storage bucket that has been created for this lab:

```
gsutil cp dq-customer-raw-data.yaml gs://Project ID-bucket
```

Copi

Task 4. Define and run a data quality job in Dataplex

The data quality process uses a data quality specification YAML file to run a data quality job and generates data quality metrics that are written to a BigQuery dataset.

In this task, you define and run a data quality job using the data quality specification YAML file uploaded to Cloud Storage in the previous task. When you define the job, you also specify a pre-created BigQuery dataset named **customer_dq_dataset** to store the data quality results.

1. In the Google Cloud Console, in the **Navigation menu** (☰), navigate to **Analytics** > **Dataplex**.

2. Under **Manage lakes**, click **Process**.

3. Click **+CREATE TASK**.

4. Under Check Data Quality, click **Create task**.

5. Enter the required information to create a new data quality job:

Property	Value
Dataplex lake	**ecommerce-lake**
Display name	`Customer Data Quality Job`
ID	Leave the default value.
Select GCS file	_____-bucket/dq-customer-raw-data.yaml
Select BigQuery dataset	_____.customers_dq_dataset
BigQuery table	`dq_results`
User service account	**Compute Engine default service account**

Leave the other default values.

Note that the Compute Engine default service account has been preconfigured for this lab to have the appropriate IAM roles and permissions. For more information, review the Dataplex documentation titled Create a service account.

6. Click **Continue**.

7. For **Start**, select **Immediately**.

8. Click **Create**.

Note: It can take several minutes for the job to run. You may need to refresh the page to see that the job has run successfully.

Task 5. Review data quality results in BigQuery

In this task, you review the tables in the **customers_dq_dataset** to identify records that are missing customer ID values or have an invalid values for emails.

1. In the Google Cloud Console, in the **Navigation menu** (☰), navigate to **BigQuery**.

2. In the Explorer pane, expand the arrow next to your project ID to list the contents: ____

3. Expand the arrow next to the **customer_dq_dataset** dataset.

4. Click on the **dq_summary** table.

5. Click on the **Preview** tab to see the results.

The **dq summary** table provides useful information about the overall data quality including the number of records that were identified to not adhere to the two rules in the data quality specification file.

6. Scroll to the last column named **failed_records_query**.

7. Click on the down arrow in the first row to expand the text and view the entire query for the **VALID_EMAIL** rule results.

Note that the query is quite long and ends with `ORDER BY _dq_validation_rule_id`.

8. Click on **+ SQL query**. Copy and paste the query into SQL Editor, and click **Run**.

The results of the query provide the email values in the **contact_info** table that are not valid.

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	_dq_validation_invocation_id	_dq_validation_rule_binding_id	_dq_validation_rule_id	_dq_validation_column_id	_dq_validation_column_value	_dq_validation_dimension
1	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	hicks	CONFORMANCE
2	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	arthur.diaz	CONFORMANCE
3	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	anthony.morton	CONFORMANCE
4	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	example.com	CONFORMANCE
5	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	sydney	CONFORMANCE
6	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	@	CONFORMANCE
7	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	example.com	CONFORMANCE
8	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	adam.neuman	CONFORMANCE
9	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	example.com	CONFORMANCE
10	fce82f48-1a38-4d88-8309-58a...	VALID_EMAIL_ID	VALID_EMAIL	email	duane	CONFORMANCE

9. Repeat steps 7-8 for the second cell that contains the query for the **VALID_CUSTOMER** rule results.

The results of the query identify that there are 10 records in the **contact_info** table that are missing ID values.

Query results

SAVE RESULTS EXPLORE DATA

JOB INFORMATION RESULTS JSON EXECUTION DETAILS EXECUTION GRAPH

Row	_dq_validation_invocation_id	_dq_validation_rule_binding_id	_dq_validation_rule_id	_dq_validation_column_id	_dq_validation_colun	_dq_validation_dimension	_dq_v
1	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
2	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
3	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
4	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
5	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
6	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
7	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
8	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
9	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false
10	fce82f48-1a38-4d88-8309-58a...	VALID_CUSTOMER	NOT_NULL	id	null	COMPLETENESS	false