Prepare Data for ML APIs on Google Cloud: Challenge Lab

Check project permissions

Before you begin your work on Google Cloud, you need to ensure that your project has the correct permissions within Identity and Access Management (IAM).

1. In the Google Cloud console, on the Navigation menu , select IAM & Admin > IAM.

2. Confirm that the default compute Service Account `{project-number}-compute@developer.gserviceaccount.com` is present and has the `editor` and `storage.admin` role assigned. The account prefix is the project number, which you can find on Navigation menu > Cloud Overview > Dashboard.

Note: If the account is not present in IAM or does not have the `storage.admin` role, follow the steps below to assign the required role.

1. In the Google Cloud console, on the Navigation menu, click Cloud Overview > Dashboard.
2. Copy the project number (e.g. `729328892908`).
3. On the Navigation menu, select IAM & Admin > IAM.
4. At the top of the roles table, below View by Principals, click Grant Access.
5. For New principals, type:

`{project-number}-compute@developer.gserviceaccount.com`

Copied!

content_copy

6. Replace `{project-number}` with your project number.
7. For Role, select Storage Admin.
8. Click Save.

Challenge scenario

As a junior data engineer in Jooli Inc. and recently trained with Google Cloud and a number of data services you have been asked to demonstrate your newly learned skills. The team has asked you to complete the following tasks.

You are expected to have the skills and knowledge for these tasks so don't expect step-by-step guides.

Task 1. Run a simple Dataflow job

In this task, you use the Dataflow batch template Text Files on Cloud Storage to BigQuery under "Process Data in Bulk (batch)" to transfer data from a Cloud Storage bucket (`gs://cloud-training/gsp323/lab.csv`). The following table has the values you need to correctly configure the Dataflow job.

You will need to make sure you have:

- Create a BigQuery dataset called `lab_141` with a table called `customers_180`.
- Create a Cloud Storage Bucket called `qwiklabs-gcp-03-2a5a831dec3a-marking`.

Field	Value
Cloud Storage input file(s)	`gs://cloud-training/gsp323/lab.csv`
Cloud Storage location of your BigQuery schema file	`gs://cloud-training/gsp323/lab.schema`
BigQuery output table	`qwiklabs-gcp-03-2a5a831dec3a:lab_141.customers_180`

Temporary directory for BigQuery loading process	`gs://qwiklabs-gcp-03-2a5a831dec3a-marking/bigquery_temp`
Temporary location	`gs://qwiklabs-gcp-03-2a5a831dec3a-marking/temp`
Optional Parameters > JavaScript UDF path in Cloud Storage	`gs://cloud-training/gsp323/lab.js`
Optional Parameters > JavaScript UDF name	`transform`
Optional Parameters > Machine Type	`e2-standard-2`

Wait for the job to finish before trying to check your progress.

1. Create BigQuery Dataset and Table

1. Go to the BigQuery section in Google Cloud Console.
2. Click on your project name (e.g., `qwiklabs-gcp-03-2a5a831dec3a`).
3. Click "Create Dataset".
    - Dataset ID: `lab_141`
    - Leave other settings as default.
    - Click "Create Dataset".
4. The table (`customers_180`) will be created automatically by the Dataflow job, so no need to create it manually.

2. Create a Cloud Storage Bucket

1. Go to the Cloud Storage section in Google Cloud Console.
2. Click "Create Bucket".
    - Name: `qwiklabs-gcp-03-2a5a831dec3a-marking`
    - Choose default options for region and storage class.
    - Click "Create".

3. Launch the Dataflow Job

1. Go to the **Dataflow** section in Google Cloud Console.

2. Click **"Create Job from Template"**.
3. Configure the job:
    - **Job Name**: e.g., `csv-to-bq-job`
    - **Region**: (use default or a nearby region)
    - **Dataflow Template**: `Text Files on Cloud Storage to BigQuery`
4. Under **"Parameters"**, enter the following:

Parameter	Value
Cloud Storage input file(s)	`gs://cloud-training/gsp323/lab.csv`
BigQuery output table	`qwiklabs-gcp-03-2a5a831dec3a:lab_141.customers_180`
Cloud Storage location of your BigQuery schema file	`gs://cloud-training/gsp323/lab.schema`
Temporary directory for BigQuery loading process	`gs://qwiklabs-gcp-03-2a5a831dec3a-marking/bigquery_temp`
Temporary location	`gs://qwiklabs-gcp-03-2a5a831dec3a-marking/temp`
JavaScript UDF path in Cloud Storage *(optional)*	`gs://cloud-training/gsp323/lab.js`
JavaScript UDF name *(optional)*	`transform`
Machine Type *(optional)*	`e2-standard-2`

5. Click **"Run Job"**.

Task 2. Run a simple Dataproc job

In this task, you run an example Spark job using Dataproc.

Example Dataproc job is shown below:

Run a Dataproc job using the values below.

1. In the Cloud Platform Console, select Navigation menu > View all products > Dataproc > Clusters, then click Create cluster.
2. Click Create for Cluster on Compute Engine.
3.

Field	Value
Dataproc Cluster	`Compute Engine`
Region	`us-central1`
Machine Series	`E2`
Manager Node	Set Machine Type to e2-standard-2
Worker Node	Set Machine Type to e2-standard-2

Max Worker Nodes	`2`
Primary disk size	`100 GB`
Internal IP only	`Deselect "Configure all instances to have only internal IP addresses`

Wait for the job to finish before trying to check your progress.

Task 2. Submit a job

To run a sample Spark job:

1. Click Jobs in the left pane to switch to Dataproc's jobs view, then click Submit job.
2. Set the following fields to update Job. Accept the default values for all other fields:

Dataproc Cluster	`Compute Engine`
Region	`Region`
Machine Series	`E2`

Manager Node	Set Machine Type to e2-standard-2
Worker Node	Set Machine Type to e2-standard-2
Max Worker Nodes	`2`
Primary disk size	`100 GB`
Internal IP only	`Deselect "Configure all instances to have only internal IP addresses`

Task 3. Use the Google Cloud Speech-to-Text API

- Use Google Cloud Speech-to-Text API to analyze the audio file `gs://cloud-training/gsp323/task3.flac`. Once you have analyzed the file, upload the resulting file to: `gs://qwiklabs-gcp-03-2a5a831dec3a-marking/task3-gcs-517.result`

1. Enable the Speech-to-Text API

- Go to the Google Cloud Console.
- Navigate to **APIs & Services > Library**.
- Search for **"Speech-to-Text API"**.
- Click **Enable**.

2. Create a Transcription Using `gcloud`

Open **Cloud Shell** and run the following command:

```
gcloud ml speech recognize-long-running \
  gs://cloud-training/gsp323/task3.flac \
  --language-code='en-US' \
  --formatting=json \
  > result.json
```

This will create a **long-running** recognition task and wait for it to complete, saving the output as `result.json`.

: Start it asynchronously

You can also submit and poll manually:

```
gcloud ml speech recognize-long-running gs://cloud-
training/gsp323/task3.flac \
  --language-code='en-US' \
  --formatting=json \
  --output=result.json
```

3. Upload the Result to Cloud Storage

Once you have the `result.json` file in your Cloud Shell environment, upload it using:

```
gsutil cp result.json gs://qwiklabs-gcp-03-2a5a831dec3a-marking/task3-gcs-
517.result
```

Task 4. Use the Cloud Natural Language API

- Use the Cloud Natural Language API to analyze the sentence from text about Odin. The text you need to analyze is "Old Norse texts portray Odin as one-eyed and long-bearded, frequently wielding a spear named Gungnir and wearing a cloak and a broad hat." Once you have analyzed the text, upload the resulting file to: `gs://qwiklabs-gcp-03-2a5a831dec3a-marking/task4-cnl-878.result`

1. Enable the Cloud Natural Language API

- Go to Cloud Console > APIs & Services > Library.
- Search for **Cloud Natural Language API** and click **Enable**.

2. Create a JSON Request File

In **Cloud Shell**, create a request file (e.g. `odin.txt`) using `nano` or any text editor:

```
nano request.json
```

Paste the following content:

```
{
  "document": {
    "type": "PLAIN_TEXT",
    "content": "Old Norse texts portray Odin as one-eyed and long-bearded,
frequently wielding a spear named Gungnir and wearing a cloak and a broad
hat."
  },
  "encodingType": "UTF8"
}
```

Press `Ctrl + O`, then `Enter` to save, and `Ctrl + X` to exit.

3. Call the Natural Language API

Run the following command to analyze **entities, syntax, and sentiment**:

```
gcloud ml language analyze-entities \
  --content-file=request.json \
  --format=json > result.json
```

Alternatively, for more comprehensive analysis, you can use:

```
gcloud ml language analyze-entities \
  --content="Old Norse texts portray Odin as one-eyed and long-bearded,
frequently wielding a spear named Gungnir and wearing a cloak and a broad
hat." \
  --format=json > result.json
```

You can also perform sentiment or syntax analysis if required.

4. Upload the Result to Cloud Storage

```
gsutil cp result.json gs://qwiklabs-gcp-03-2a5a831dec3a-marking/task4-cnl-
878.result
```