fourcast.

Google Cloud Platform

Data Workshop

08/05/2020

0 Google Cloud Platform

0.1 Go the the console: https://console.cloud.google.com/home
0.2 Select the project **rbfa-workshop1** in the top blue bar

1 Cloud Storage

1.1 Make your own bucket

1.1.1 Go to Storage in the console, in the project **rbfa-workshop1**
1.1.2 Click *Create Bucket* and apply the following:

- **Name**: rbfa-workshop1-{your_name}
- **Location type**: regional
- **Location**: europe-west1
- **Storage class**: Standard
- **Access-control**: Fine-grained
- Leave the rest to its default

1.2 Upload a file to your bucket

1.2.1 Download the RBFA logo to your local computer:
1.2.2 Go to your bucket in the console
1.2.3 Click *Upload Files* and upload the file
1.2.4 Verify that the file is uploaded

1.3 Change the permissions of the file

1.3.1 Go to the file in Cloud Storage
1.3.2 Click on “Edit Permissions”
1.3.3 Click on “Add Item” and apply the following:

- **Entity**: Public
- **Name**: allUsers
- **Access**: Reader

⚠️ The file is now publicly accessible to anyone on the internet.
1.3.4 Verify that the file now has a public link

1.4 Delete the file using Cloud Shell - **gsutil**

1.4.1 Start Cloud Shell
1.4.2 Run the following command

```
gsutil rm gs://{your_bucket_name}/{path_to_file}
```

1.4.3 Verify that the file is deleted

1.5 Copy a CSV file using Cloud Shell from another bucket

1.5.1 Start Cloud Shell
1.5.2 Run the following command (in one line)

```
gsutil cp gs://rbfa-workshop1-data/premier-league/season-1819.csv
gs://{your_bucket_name}/
```

1.5.3 Verify that the file is copied

1.6 Enable object versioning for the bucket

1.6.1 Start Cloud Shell
1.6.2 Run the following command

```
gsutil versioning set on gs://{your_bucket_name}
```

1.6.3 Verify that versioning is enabled by running the following command

```
gsutil versioning get gs://{your_bucket_name}
```

1.7 List versions

1.7.1 Start cloud shell
1.7.2 Run the following command

```
gsutil ls -a gs://{your_bucket_name}
```

1.7.3 Copy the file again by repeating step 1.5.2
1.7.5 Verify that there are two versions of the file season-1819.csv by repeating step 1.7.2

1.8 Create lifecycle rules

1.8.1 Go to the Storage browser in the console, where the buckets are listed
1.8.2 For your bucket, click on *None* in the column “Lifecycle Rules”
1.8.3 Create a lifecycle rule that will set the storage class of a file to “Nearline” if the age of a file is 365 days and the storage class of the file is “Standard”
1.8.4 Create another lifecycle rule that will delete a file if it has more than 5 newer versions

2 BigQuery

2.1 Make your own dataset

2.1.1 Go to BigQuery in the console
2.1.2 Click on the project **rbfa-workshop1** in the left panel
2.1.3 Click on “Create Dataset” and apply the following:

- **Dataset ID**: PREMIER_LEAGUE_{YOUR_NAME}
- **Data Location**: EU
- **Default table expiration:** 10 days
- Leave the rest to its default

2.2 Create a table

2.2.1 Go to your dataset in the console
2.2.2 Click on *Create Table* and apply the following

- **Source**
  - **Create table from:** Google Cloud Storage
  - **File:** browse for the csv in your bucket: {your_bucket_name}/premier-league/season-1819.csv
- **Destination**
  - **Project:** rbfa-workshop1
  - **Dataset:** {your_dataset}
  - **Table type:** Native table
  - **Table name:** SEASON_1819
- **Schema**
  - ☑Auto detect schema and input parameters

  Leave the rest to its default

2.2.3 Verify that the table is created
2.2.4 Check the table details by using “Details”
2.2.5 Explore the data by using “Preview”

2.3 Run a query

2.3.1 Run the following query in the query editor to list all the referees: (column explanations)

```
SELECT DISTINCT Referee FROM `{project_id}.{dataset_id}.SEASON_1819`
ORDER BY Referee
```

2.3.2 Try to figure out which referee gave the most yellow cards on average per game

Referee: Referee name
HY: Home Team Yellow Cards
AY: Away Team Yellow Cards

Solution:

```

```

2.4 Share the dataset

2.4.1 Go to your dataset in the console
2.4.2 Click on “Share Dataset”
2.4.3 In “Add members”, type your e-mail address and add yourself as a “Bigquery Data Viewer”
Note: Since you were the person that created the dataset, you automatically have the “BigQuery Data Owner” role for that dataset
2.4.4 Verify that you are listed as a “BigQuery Data Viewer”

2.5 Delete the table using Cloud Shell - **bq**

2.5.1 Start Cloud Shell
2.5.2 Run the following command:

```
bq rm {project_id}:{dataset_id}.{table_id}
```

2.5.3 Confirm by typing `Y`
2.5.4 Verify that the table is deleted

3 Cloud Functions

3.1 Create your own HTTP Cloud Function

3.1.1 Go to Cloud Functions in the console
3.1.2 Click *Create Function* and apply the following:

- **Name:** hello-{your_name}
- **Region:** europe-west1
- **Memory allocated:** 128MiB
- **Trigger:** HTTP
- **Authentication:** ☑Allow unauthenticated invocations

  ⚠️ This function is now publicly accessible to anyone on the internet.
- **Source Code:** Inline editor
- **Runtime:** Python 3.7
- **MAIN.PY:** Change `'Hello World!'` to `'Hello {your_name}!'` in line 16

3.1.3 Wait until the function is deployed
3.1.4 Test the function by going to the following url in your browser:
https://europe-west1-{project_id}.cloudfunctions.net/{function_name}

3.2 Delete the function using Cloud Shell - **gcloud**

3.2.1 Start Cloud Shell
3.2.2 Run the following command:

```
gcloud functions delete {function_name} --region europe-west1
```

3.2.3 Confirm by typing `Y`
3.2.4 Verify that the function is deleted

3.3 Create a Storage-triggered Cloud Function

This function will be triggered on every file that is created in your bucket, it will then create a BigQuery table in your dataset from the data in the file.

3.3.1 Go to Cloud Functions in the console

3.3.2 Click *Create Function* and apply the following:

- **Name:** storage-to-bigquery-{your_name}
- **Memory allocated:** 256MiB
- **Trigger:** Cloud Storage
    - **Event type:** Finalize/Create
    - **Bucket:** {your_bucket}
- **Source Code:** Inline editor
- **Runtime:** Python 3.7
- **MAIN.PY:**

```
def gcs_to_bq(event, context):
    import os
    from google.cloud import bigquery
    dataset_id = os.getenv('DATASET')
    project_id = os.getenv('GCP_PROJECT')
    bucket = event['bucket']
    file_name = event['name']
    table_name = file_name.split('/')[-1].split('.')[0].upper().replace('-',
'_')
    table_id = f'{project_id}.{dataset_id}.{table_name}'
    bq_client = bigquery.Client(project_id)
    job_config = bigquery.LoadJobConfig()
    job_config.autodetect = True
    job_config.write_disposition = 'WRITE_TRUNCATE'
    load_job = bq_client.load_table_from_uri(
        f'gs://{bucket}/{file_name}',
        table_id,
        job_config=job_config
    )
    load_job.result()
    print(f'Created the table {table_id} from the file {file_name}.')
```

- **REQUIREMENTS.TXT:**

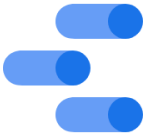
```
google-cloud-bigquery
```

- **Function to execute:** gcs_to_bq
- **Advanced options**
    - **Region:** europe-west1
    - **Environment variables:**
        - **Name:** DATASET **Value:** {your_dataset}

3.3.3 Test the function by copying all the CSVs for the past 10 premier league seasons to your bucket, by running the following command in Cloud Shell (in one line)

```
gsutil -m cp gs://rbfa-workshop1-data/premier-league/*
gs://{your_bucket}/premier-league/
```

3.3.4 View the logs of your function and verify that the tables are created in BigQuery



4 Data Studio

4.1 Create a BigQuery authorized view in a separate dataset

4.1.1 Go to BigQuery in the console
4.1.2 Create a new dataset, in the EU, called DATA_MARTS_{YOUR_NAME}
4.1.3 Run the following query in the query editor

```
SELECT Date, HomeTeam, AwayTeam, FTHG, FTAG, FTR, Referee, HS,`AS`, HST,
AST, HF, AF, HC, AC, HY, AY, HR, AR
FROM `{project_id}.{premier_league_dataset_id}.*`
```

4.1.4 Click on “Save View” and apply the following:

- **Project name:** rbfa-workshop1
- **Dataset name:** {your_data_marts_dataset}
- **Table name:** PREMIER_LEAGUE

4.1.5 Go to the source dataset and click on “Share Dataset”
4.1.6 Go to “authorized views” and add your view

4.2 Explore the data using Data Studio

4.2.1 Run the following query

```
SELECT DATE, HomeTeam AS TEAM, FTHG AS GOALS, HY AS YELLOW_CARDS FROM
`{project_id}.{data_marts_dataset_id}.PREMIER_LEAGUE`
UNION ALL
SELECT DATE, AwayTeam AS TEAM, FTAG AS GOALS, AY AS YELLOW_CARDS FROM
`{project_id}.{data_marts_dataset_id}.PREMIER_LEAGUE`
```

4.2.1 Click on “Explore Data”
4.2.2 Click on “Explore with DataStudio”
4.2.3 Delete the automatically generated table
4.2.4 Create a “Combo Chart” that shows total number of goals and yellow cards per team by applying the following:

- **Dimension:** TEAM
- **Metric:**
  - GOALS
  - YELLOW_CARDS

4.2.5 Make sure that the bars represent the GOALS and the line represents the YELLOW_CARDS and that the bars are sorted ascending
4.2.6 Change the metrics so that they represent the average goals/cards per game instead of the totals

- Click on the metric
- Select “Create Field”
- Provide a name: e.g. Avg goals per game
- Provide a formula: e.g. SUM(GOALS) / Record Count

4.2.7 Add a filter for the date and only show result for 2015
4.2.8 Add a filter for the teams and only show the results of Chelsea, Man City and Liverpool