

# 用Cloud Functions导入数据到BigQuery

Author: [eugeneyu@google.com](mailto:eugeneyu@google.com)

首先在谷歌云控制台创建一个Cloud Functions函数。参照下面截图填写相应配置。其中“Bucket”选择源桶。“Event Type”选择“Finalize/Create”。

Cloud Functions    Create function

1 Configuration — 2 Code

### Basics

Function name \*  
load\_from\_gcs\_to\_bq

Region  
us-central1

### Trigger

#### Cloud Storage

Trigger type  
Cloud Storage

Event type \*  
Finalize/Create

Bucket \*  
auto\_load\_test    BROWSE

☐ Retry on failure

SAVE    CANCEL

VARIABLES, NETWORKING AND ADVANCED SETTINGS

在下面的高级配置中，适当增大Cloud Functions的分配内存和超时设置（最长9分钟）。当导入的文件较大时，配置大一些的内存和超时可以防止出错。

VARIABLES, NETWORKING AND ADVANCED SETTINGS

ADVANCED ENVIRONMENT VARIABLES CONNECTIONS

Memory allocated \*  
1 GiB

Timeout \*  
540 seconds ?

Maximum function instances ?

Service account  
App Engine default service account ?

在环境变量配置中创建新的环境变量，指定导入文件的特定前缀和BigQuery的目的表。如果有多  
个不同前缀文件，要分布导入不同的目的表，可以用相同的顺序列表填写。函数代码会按照列表  
顺序，将不同前缀文件数据导入到不同的BigQuery目的表。

- Name: GCS\_FILE\_PREFIX
- Value: 需要导入的文件内的关键字前缀，如“user\_id,produc\_id”
- Name : BQ\_TABLE\_IDS
- Value : 目的表名列表，用逗号隔开，表名格式为  
`your-project.your_dataset.your_table_name`

VARIABLES, NETWORKING AND ADVANCED SETTINGS ^

ADVANCED ENVIRONMENT VARIABLES CONNECTIONS

Build environment variables ?

+ ADD VARIABLE

Runtime environment variables ?

| Name            | Value                  |
|-----------------|------------------------|
| BQ_TABLE_IDS    | testbq.eugene_ds.dwd_u |
| GCS_FILE_PREFIX | part-00000,part-00001  |

+ ADD VARIABLE

在下一步配置函数代码。首先运行时选择Python 3.7，Entry Point是初始调用的函数，选择代码里的入口函数名。

因为上面“Source code”选择了“Inline editor”，所以可以用内建的网页代码编辑器输入代码。将示例代码填入“main.py”标签的代码输入框。

Cloud Functions

Edit function

Configuration

Code

Runtime

Python 3.7

Entry point

load\_file

Source code

Inline Editor

main.py

requirements.txt

```
1 from google.cloud import storage
2 from google.cloud import bigquery
3 import os
4 import sys
5 import logging
6 from flask import abort
7
8 #BQ_TABLE_ID = os.environ['BQ_TABLE_ID']
9 BQ_TABLE_IDS = os.environ['BQ_TABLE_IDS']
10 GCS_FILE_PREFIX = os.environ['GCS_FILE_PREFIX']
11
12 # Construct a BigQuery client object.
13 client = bigquery.Client()
14
15 def exit_abort():
16     return abort(500)
17
18 def load_file(event, context):
19
20     bq_table_ids = BQ_TABLE_IDS.split(',')
21     gcs_file_prefix = GCS_FILE_PREFIX.split(',')
22
23     if len(bq_table_ids) != len(gcs_file_prefix):
24         error_msg = 'BQ_TABLE_IDS has different size ({} than GCS_FILE_PREFIX ({}).\n'.format(
25             len(bq_table_ids), len(gcs_file_prefix))
26         sys.stderr.write(error_msg)
27         logging.error(RuntimeError(error_msg))
28         exit_abort()
29
30     bucket_name = event['bucket']
31     object_name_src = event['name']
32
33     i = 0
34     for file_prefix in gcs_file_prefix:
35         if file_prefix in object_name_src:
36             print(object_name_src + " -- " + file_prefix + " -- " + bq_table_ids[i])
37             import_to_bq(bucket_name, object_name_src, bq_table_ids[i])
38             break
39             i += 1
40
41
42 def import_to_bq(bucket_name, object_name_src, table_id):
43 ..
```

PREVIOUS

DEPLOY

CANCEL

本例的代码可以在Github上查看。地址是

[https://github.com/eugeneyu/cloud-demos/blob/master/cloud-functions/load\\_from\\_gcs\\_to\\_bq.py](https://github.com/eugeneyu/cloud-demos/blob/master/cloud-functions/load_from_gcs_to_bq.py)

也可以从下面拷贝。

```
from google.cloud import storage
from google.cloud import bigquery
import os
import sys
import logging
from flask import abort

BQ_TABLE_IDS = os.environ['BQ_TABLE_IDS']
```

```

GCS_FILE_PREFIX = os.environ['GCS_FILE_PREFIX']

# Construct a BigQuery client object.
client = bigquery.Client()

def exit_abort():
    return abort(500)

def load_file(event, context):

    bq_table_ids = BQ_TABLE_IDS.split(',')
    gcs_file_prefix = GCS_FILE_PREFIX.split(',')

    if len(bq_table_ids) != len(gcs_file_prefix):
        error_msg = 'BQ_TABLE_IDS has different size ({{}}) than
GCS_FILE_PREFIX ({{}})\n'.format(
            len(bq_table_ids), len(gcs_file_prefix))
        sys.stderr.write(error_msg)
        logging.error(RuntimeError(error_msg))
        exit_abort()

    bucket_name = event['bucket']
    object_name_src = event['name']

    i = 0
    for file_prefix in gcs_file_prefix:
        if file_prefix in object_name_src:
            print(object_name_src + " -- " + file_prefix + " -- " +
bq_table_ids[i])
            import_to_bq(bucket_name, object_name_src, bq_table_ids[i])
            break
        i += 1

def import_to_bq(bucket_name, object_name_src, table_id):

    job_config =
bigquery.LoadJobConfig(source_format=bigquery.SourceFormat.OCR)

    destination_table = client.get_table(table_id)

    row_num_before_load = destination_table.num_rows

```

```

# uri is like
"gs://cloud-samples-data/bigquery/us-states/us-states.orc"
uri = "gs://" + bucket_name + "/" + object_name_src

load_job = client.load_table_from_uri(
    uri, table_id, job_config=job_config
) # Make an API request.

load_job.result() # Waits for the job to complete.

destination_table = client.get_table(table_id)
row_num_after_load = destination_table.num_rows
print("Loaded {} rows.".format(row_num_after_load-row_num_before_load))

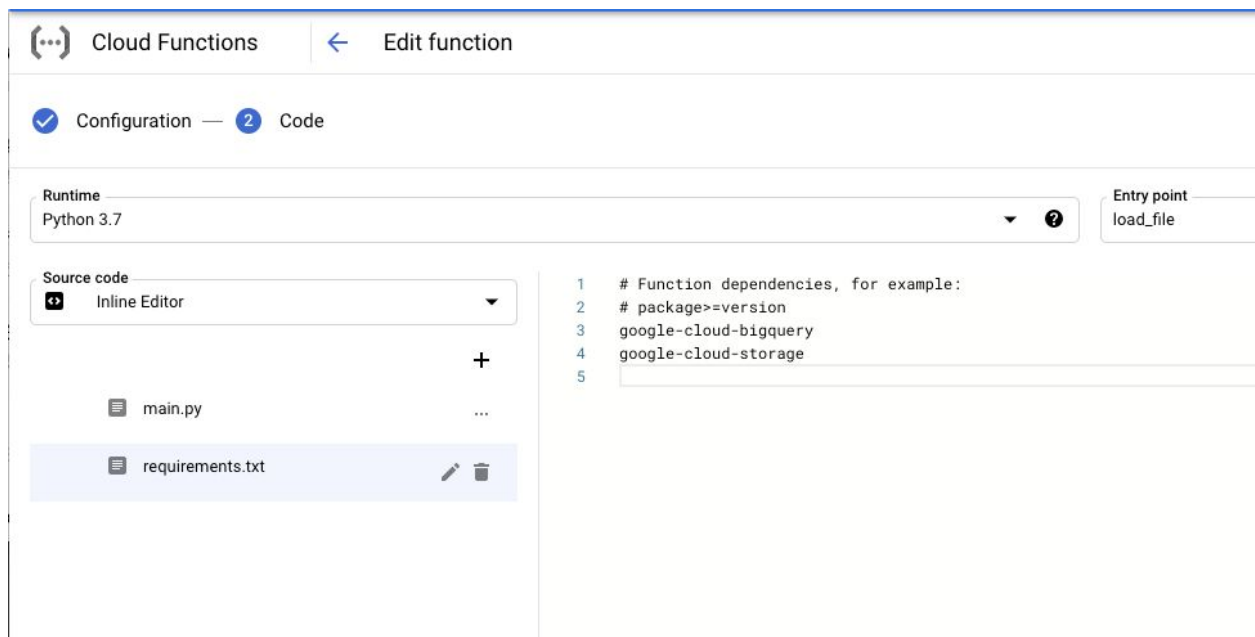
```

然后在“requirements.txt”标签输入框加上两行，指定要用到的外部库：

```

google-cloud-bigquery
google-cloud-storage

```



然后点击部署，等待几分钟可以看到函数已经部署成功。

