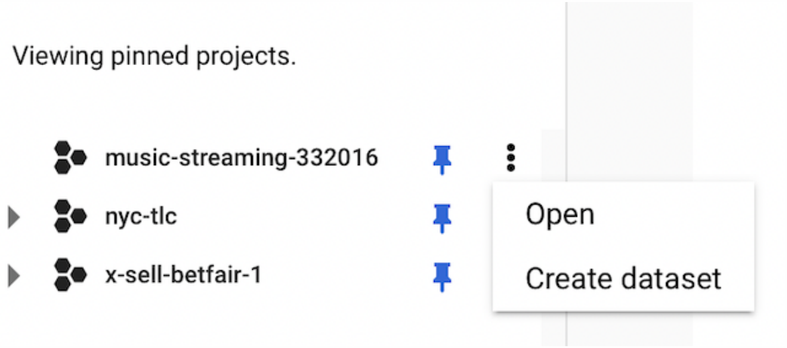
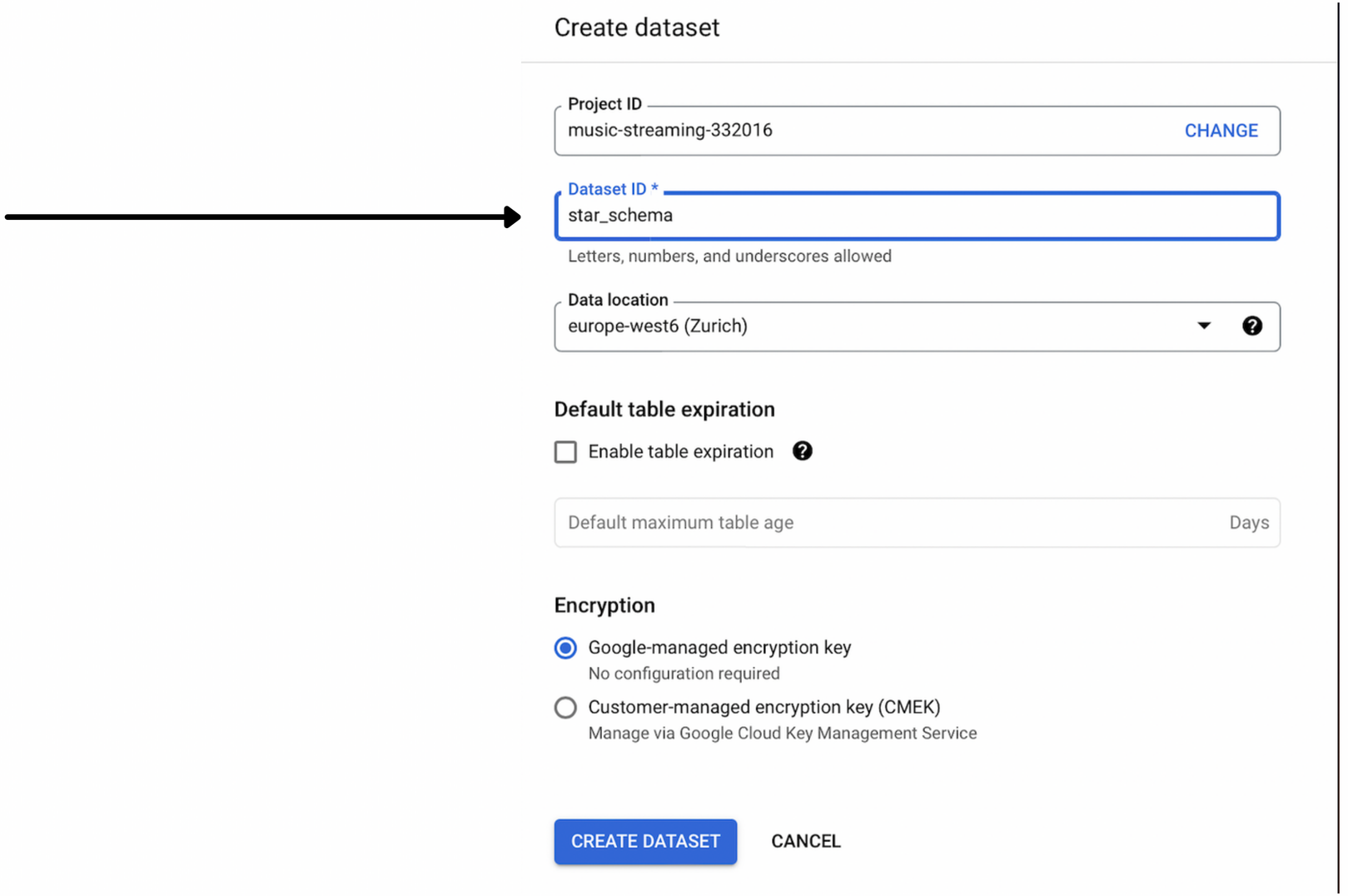

PADDYPOWER.
betfair

Crearea obiectelor in BQ

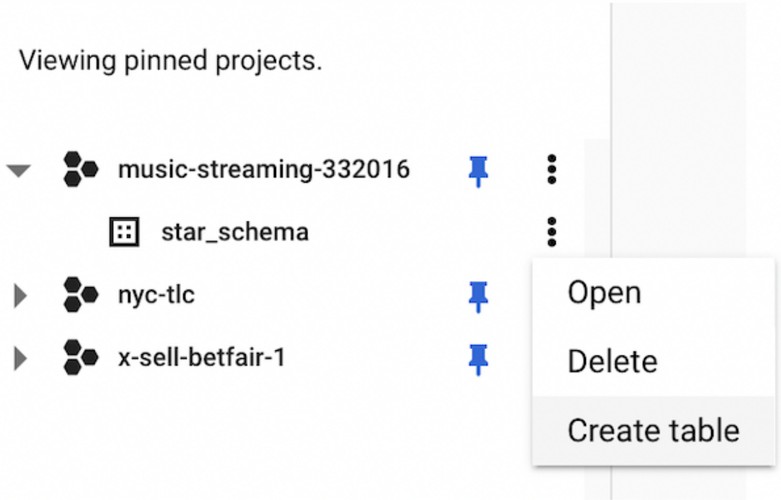
- Proiect nou în Google Cloud Platform
- Dataset nou în BigQuery
  - Tabele noi în dataset -> pe baza de schemă sau completarea fiecărei coloane
- (Optional) Un bucket nou in Google Storage
- Popularea tabelelor cu date de test
  - Mockaroo -> încărcare fișiere csv în Google Storage
  - Faker -> generare date de test cu ajutorul librăriei Faker
  - BigQuery API -> import data cu ajutorul API-ului (Python)

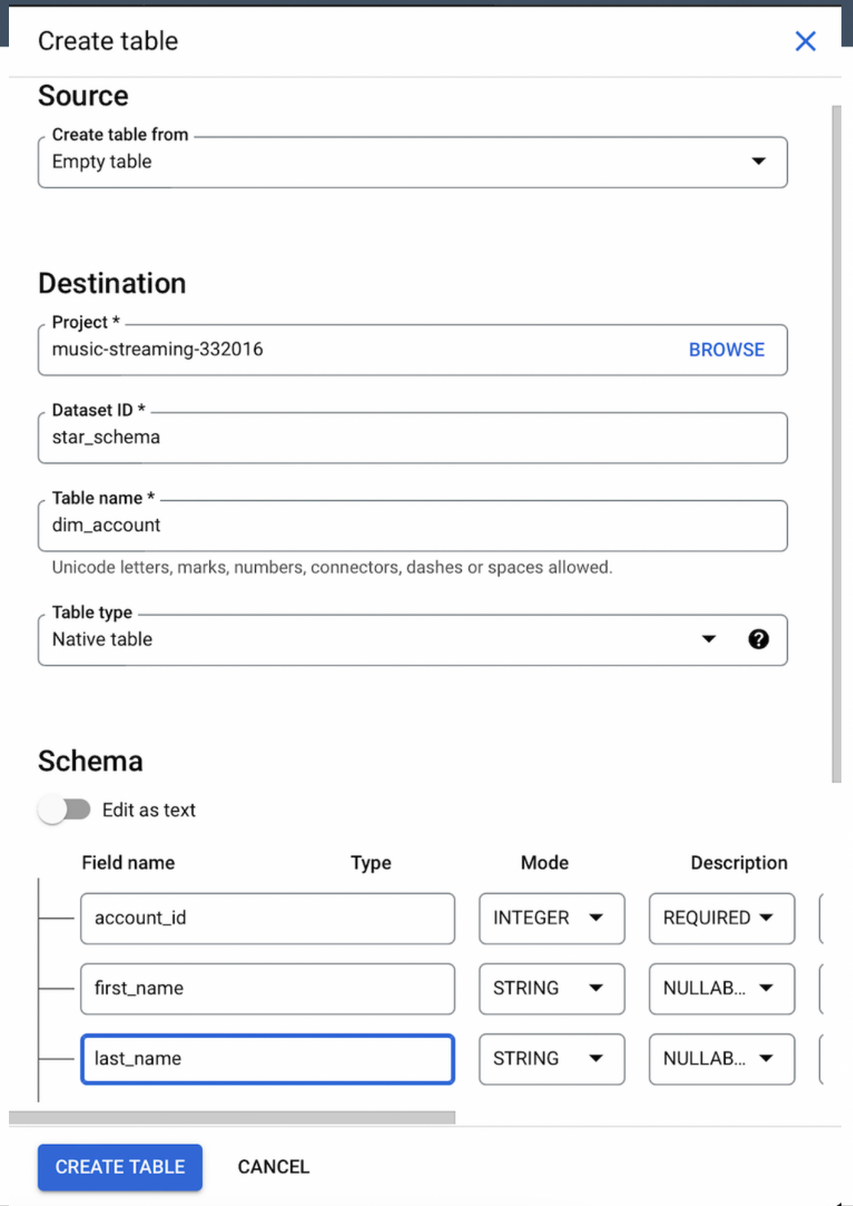
Crearea obiectelor in BQ





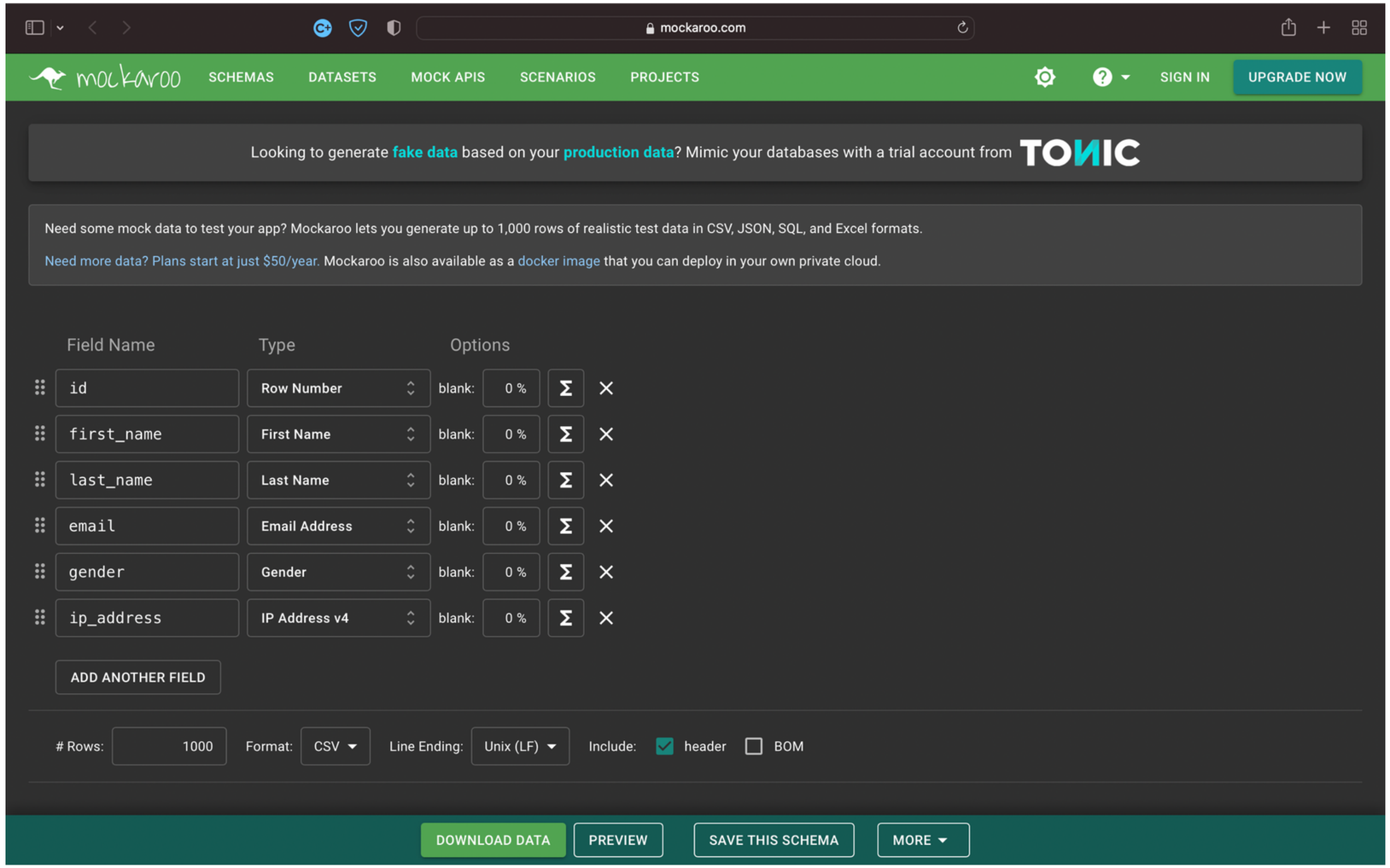
Crearea obiectelor in BQ

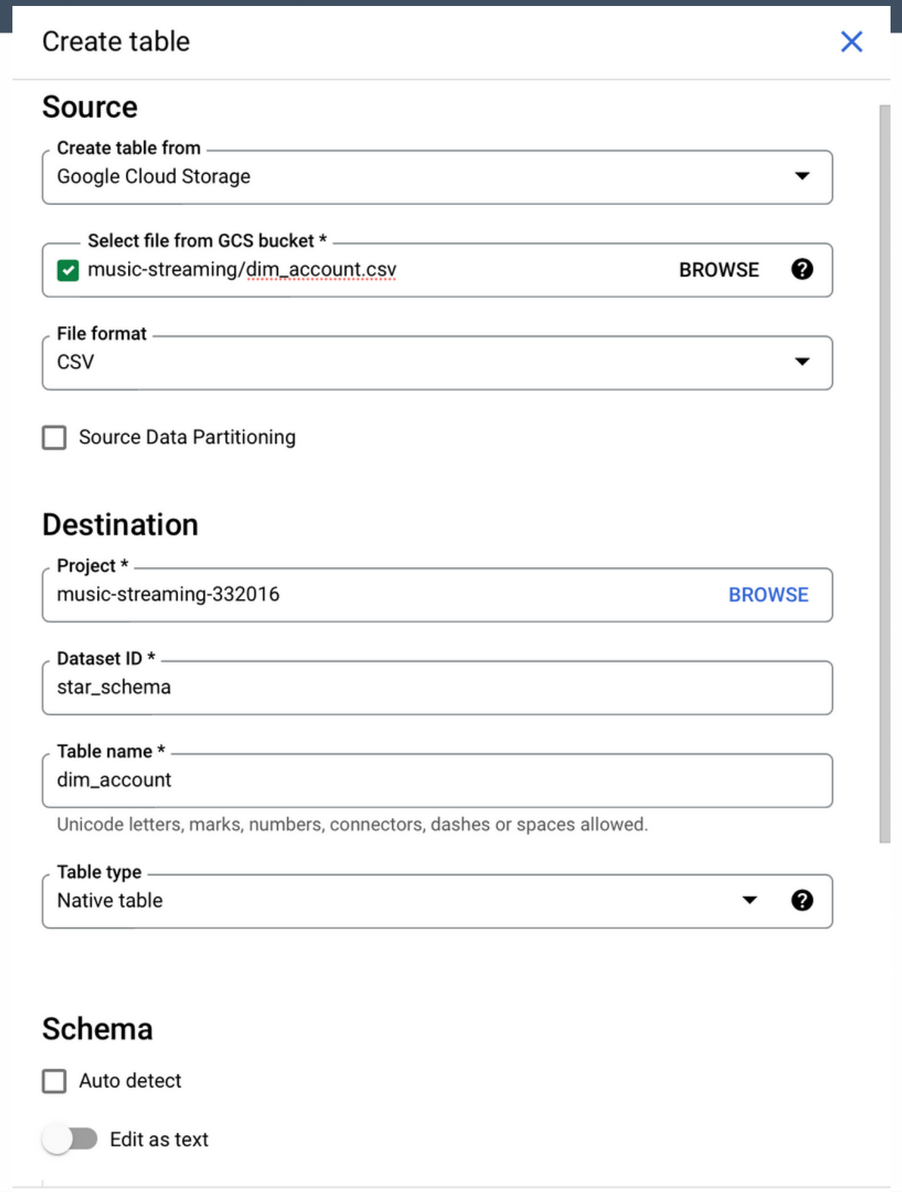




Populatea tabelelor

Optiunea 1: Mockaroo + încărcare fișier csv in Google Storage

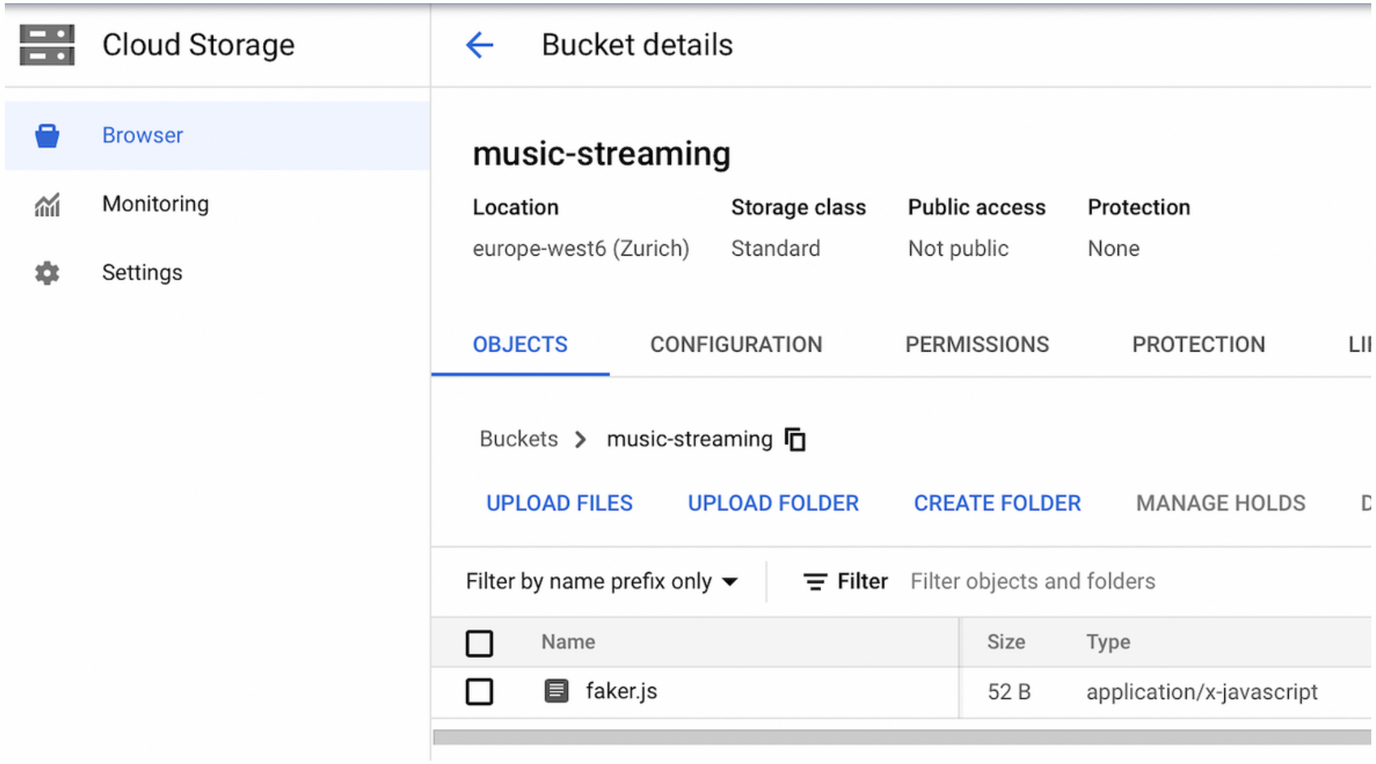




Popularea tabelelor

Opțiunea 2: generarea datelor de test cu ajutorul librăriei Faker

În acest caz, trebuie să încarcăm un fișier ajutător in Google Storage care conține librăria și să creăm funcțiile de generare de date.



Popularea tabelelor

Opțiunea 2: generarea datelor de test cu ajutorul librăriei Faker

RUN SAVE SCHEDULE MORE

```
/** Generator function */
CREATE TEMP FUNCTION dummy_account()
RETURNS STRUCT<account_id Int,
               full_name String,
               email_address String,
               date_of_birth Timestamp,
               gender String,
               address Struct<street_name String,
                              street_number Int,
                              city String
                              >,
               country String
               >
LANGUAGE js
AS """
var faker = getFaker()
var dummy_account = {};

dummy_account.account_id = faker.random.number();
dummy_account.full_name = faker.name.findName();
dummy_account.email_address = faker.internet.email();
dummy_account.date_of_birth = faker.date.past();
let genders = [ 'F' , 'M' ];
dummy_account.gender = faker.random.arrayElement(genders);
dummy_account.country = faker.address.country();

```

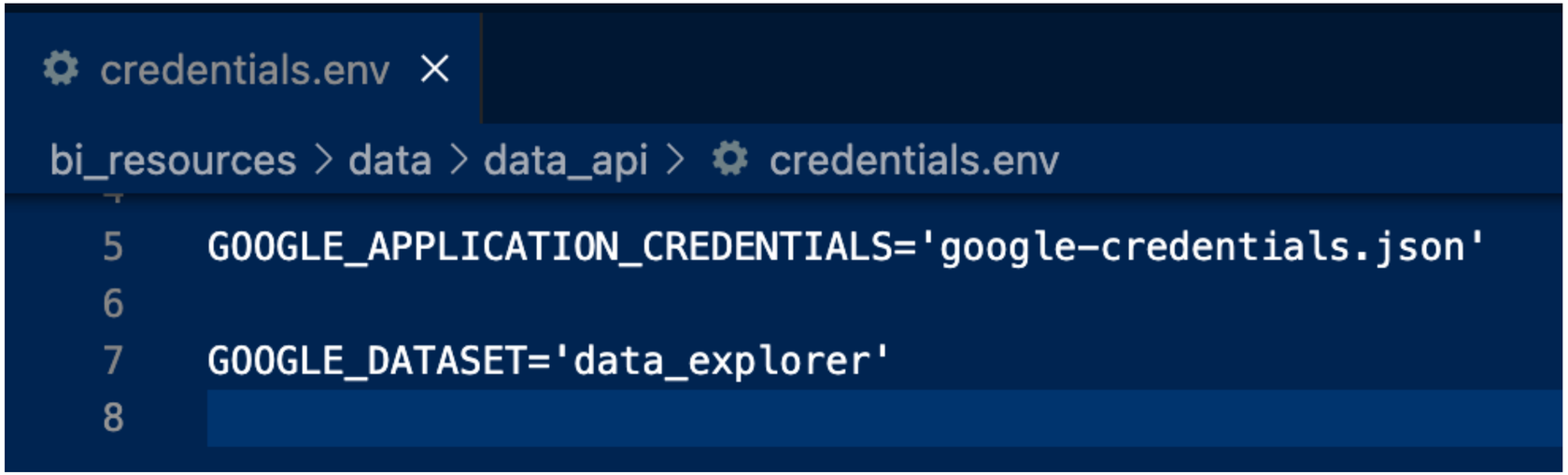
```
/** Insert 1000 dummy records */
INSERT INTO
`music-streaming-332016.star_schema.dim_account`
(SELECT dummy_account().*  from UNNEST(GENERATE_ARRAY(1, 1000)));

select * from `music-streaming-332016.star_schema.dim_account` ;
```

Generarea datelor de test

Opțiunea 3: BigQuery API (Python)

- Avem nevoie de un Service Account pentru access
  IAM & Admin -> Service Accounts -> Create Service Account

- Detaliile Service Account-ului le păstrăm în proiectul local
- Într-un fișier .env îi atribuim un nume pe care apoi îl putem referenția



Generarea datelor de test

Opțiunea 3: BigQuery API (Python)

```
from google.cloud import bigquery
from dotenv import load_dotenv
import os



# load global variables
load_dotenv(dotenv_path='credentials.env')

GOOGLE_APPLICATION_CREDENTIALS = os.getenv('GOOGLE_APPLICATION_CREDENTIALS')
GOOGLE_DATASET = os.getenv('GOOGLE_DATASET')

client = bigquery.Client()
dataset_ref = client.dataset(GOOGLE_DATASET)
table_ref = dataset_ref.table(table)
client.load_table_from_dataframe(dataframe, table_ref).result()
```

Calitatea datelor

Defectele datelor pot fi corectate înainte sau chiar în timpul raportării.
Acest lucru este posibil folosind pași intermediari în procesarea datelor.

- Tool-uri externe - teste, verificări, etc
- Verificări interne - cod, loguri sau metrici

Calitatea datelor este foarte importantă

- Detectarea încercărilor de fraudă
- Detectarea încercărilor de spălare de bani
- Protejarea clienților

Calitatea datelor

Calitatea datelor are mai multe caracteristici:

- Acuratețe și precizie
- Validitate și legitimitate
- Consistență și fiabilitate
- Relevanță și actualitate
- Completitudine și exhaustivitate
- Disponibilitate și accesibilitate
- Granularitate și unicitate

Calitatea datelor

Este important să verificăm calitatea datelor cu care vom lucra mai departe.
De exemplu, pentru fiecare cont ar trebui să existe o singură intrare în baza de date.

```
SELECT
  COUNT(*),
  full_name,
  email_address
FROM
  `music-streaming-332016.star_schema.dim_account`
GROUP BY 2, 3
HAVING COUNT(*) > 1;
```

Calitatea datelor

```
SELECT
  COUNT(*),
  song_title
FROM
  `music-streaming-332016.star_schema.dim_song`
GROUP BY 2
HAVING COUNT(*) > 1;
```

essing location: us-west1

uery results SAVE RESULTS EXPLORE DATA

uery complete (0.3 sec elapsed, 415.9 KB processed)

ob information Results JSON Execution details

f0_	song_title
2	delectus quia eos
2	voluptatem excepturi itaque
2	quia et accusantium
2	est et voluptas
2	excepturi a reprehenderit

Același lucru e valabil și pentru celelalte tabele.

Este corectă aceasta verificare?

Calitatea datelor

```
SELECT
  s.stream_id,
  a.full_name,
  a.country,
  song.song_title,
  d.date
FROM
  `music-streaming-332016.star_schema.fact_streams` s
JOIN
  `music-streaming-332016.star_schema.dim_account` a
ON
  a.accout_id = s.account_id
JOIN
  `music-streaming-332016.star_schema.dim_song` song
ON
  CAST (s.song_id AS integer) = song.song_id
JOIN
  `music-streaming-332016.star_schema.dim_date` d
ON
  s.stream_date_id = d.date_id
WHERE
  date = '2019-06-05'
  AND full_name = 'Karli Johnston';
```

Cum putem interpreta acest rezultat?

Row	stream_id	full_name	country	song_title	date
1	51431	Karli Johnston	Romania	distinctio dolore et	2019-06-05
2	51431	Karli Johnston	Romania	distinctio dolore et	2019-06-05

Calitatea datelor

- Defectele datelor pot fi corectate înainte sau chiar la pasul de raportare
- Acest lucru este posibil folosind Tableau Prep (prepocesare / pregatire)
- Sau folosind direct funcții în BigQuery
- Sau pre-procesând datele înaintea încărcării (ex. Pandas)

Toate resursele prezentate sunt disponibile aici:

https://github.com/autrefois/bi_resources