PERCONA

Databases run better with Percona

PERCONA

How To Generate Test Data for Your Database Project

I'm Mario García

Technical Evangelist at Percona

mariogmd

mariogmd

mario.garcia@percona.com

Agenda

- Dependencies
- Database
- Directory Structure
- Fake Data with Faker
- Providers and Properties
- Creating a Pandas DataFrame
- Connection to the Database
- Database Schema Definition
- What is Multiprocessing?
- Generating Data

Dependencies

requirements.txt

```
pandas
sqlalchemy
tqdm
faker
```

environment.yml

```
name: percona
dependencies:
  - python=3.10
  - pandas
  - sqlalchemy
  - tqdm
  - faker
```

Dependencies

requirements.txt

```
...
PyMySQL
psycopg2
pymongo
```

pip install -r requirements.txt

environment.yml

```
...
  - PyMySQL
  - psycopg2
  - pymongo
```

conda env create -f environment.yml

Database

- MySQL & PostgreSQL

create database company;

- MongoDB
  - DB: *company*
  - Collection: employees

Directory Structure

- modules/
  - base.py
  - dataframe.py
  - schema.py

- environment.py
- mongodb.py
- requirements.py
- sql.py

Fake Data with Faker

```
from faker import Faker

fake = Faker()
for _ in range(10):
    print(fake.name())
```

Sharon Deleon

Tiffany Nelson

Manuel Ramos

Patricia Mendoza

Sara Barrett

Daniel Sanchez

Jeffery Thomas

Clarence Salinas

Nicole Henry

Mark Bond

Providers and Properties

- faker.providers.person
  - name → John Doe
  - first_name → Katherine
  - last_name → Chang

- faker.providers.address
  - address → 791 Crist Parks, Sashabury, IL 86039-9874
  - city → Sashabury
  - country → Hungary

Providers and Properties

- faker.providers.job
  - job → Musician
- faker.providers.company
  - company → Acme Ltd

- faker.providers.internet
  - email → achang@green.info
  - company_email → juancampos@example.net

Creating a Pandas DataFrame

```
from multiprocessing import cpu_count
import pandas as pd
from tqdm import tqdm
from faker import Faker
```

- pandas
- tqdm()
- faker()
- cpu_count()

Creating a Pandas DataFrame

```
fake = Faker()

num_cores =
cpu_count() - 1
```

- Creates and initializes a faker generator
- Number of cores minus one

Columns

- first_name
- last_name
- job
- company
- address
- city
- country
- email

Creating a Pandas DataFrame

```
def create_dataframe(arg):
    x = int(60000/num_cores)
    data = pd.DataFrame()
    for i in tqdm(range(x), desc='Creating DataFrame'):
        data.loc[i, 'first_name'] = fake.first_name()
        data.loc[i, 'last_name'] = fake.last_name()
        data.loc[i, 'job'] = fake.job()
        data.loc[i, 'company'] = fake.company()
        data.loc[i, 'address'] = fake.address()
        data.loc[i, 'city'] = fake.city()
        data.loc[i, 'country'] = fake.country()
        data.loc[i, 'email'] = fake.email()
    return data
```

Connection to the Database

MySQL and PostgreSQL

```
from sqlalchemy import create_engine

from sqlalchemy.orm import sessionmaker
```

MySQL

```
engine = create_engine("mysql+pymysql://user:password@localhost/company")

Session = sessionmaker(bind=engine)
```

Connection to the Database

PostgreSQL

```
engine = create_engine("postgresql+psycopg2://user:password@localhost:5432/company")

Session = sessionmaker(bind=engine)
```

MongoDB

```
from pymongo import MongoClient

uri = "mongodb://user:password@localhost:27017/"

client = MongoClient(uri)
```

Database Schema Definition

```
from sqlalchemy.types import *

schema = {
    "first_name": String(50),
    "last_name": String(50),
    "job": String(100),
    "company": String(100),
    "address": String(200),
    "city": String(100),
    "country" String(100),
    "email": String(50)
}
```

What is Multiprocessing?

PERCONA

Generating Data

```
from multiprocessing import Pool

from multiprocessing import cpu_count

import pandas as pd
```

```
from modules.dataframe import create_dataframe

from modules.schema import schema *

from modules.base import Session, engine *

from modules.base import client **
```

Generating Data

```
if __name__ == "__main__":
    num_cores = cpu_count() - 1
    with Pool() as pool:
        data = pd.concat(pool.map(create_dataframe, range(num_cores)))
```

MySQL and PostgreSQL

```
    data.to_sql(name='employees', con=engine, if_exists = 'append', index=False, dtype=schema)
```

Generating Data

MongoDB

```
data_dict = data.to_dict('records')
db = client["company"]
collection = db["employees"]
collection.insert_many(data_dict)
```

Generating Data

MySQL

```
    with engine.connect() as conn:
        conn.execute("ALTER TABLE employees ADD id INT NOT NULL AUTO_INCREMENT PRIMARY KEY FIRST;")
```

PostgreSQL

```
    with engine.connect() as conn:
        conn.execute("ALTER TABLE employees ADD COLUMN id SERIAL PRIMARY KEY;")
```

Demo

1. Download Anaconda → https://www.anaconda.com/products/distribution
2. Install Anaconda

   $ bash Anaconda3-2022.05-Linux-x86_64.sh

3. Clone repository

   $ git clone https://github.com/mattdark/data-generator

4. Change to the data-generator directory

   $ cd data-generator

Demo

5. Configure Python environment

   `$ conda env create -f environment.yml`

6. Edit modules/base.py

7. Run the script

   `$ python sql.py`

8. Check your database

Resources

- Faker
- Faker - Bundled Providers
- Faker - Community Providers
- SQLAlchemy

- Schema Definition Language
- Multiprocessing
- Using Multiprocessing to Make Python Code Faster

PERCONA

We're Hiring

https://www.percona.com/about/careers

Thank You!