Advanced databases

Project documentation

Bartosz Bywalec

Mateusz Motyka

Dataset description:

CATEGORY	SUBCATEGORY	BRAND	SLOGAN
Apparel slogans	Beachwear & swimwear slogans	Speedo Swimwear	Fueled by water.
Apparel slogans	Beachwear & swimwear slogans	Speedo Swimwear	Speedo. Born in the water.
Apparel slogans	Beachwear & swimwear slogans	TA-BOU Beachwear	Beachwear competence.
Apparel slogans	Beachwear & swimwear slogans	Arena, sport swimwear, gear for swimming	Arena. Water instinct.
Apparel slogans	Beachwear & swimwear slogans	Arena, sport swimwear, gear for swimming	What it takes to win.
Apparel slogans	Beachwear & swimwear slogans	Arena Bodylift, body shaping swimsuits	Beauty in all its forms.
Apparel slogans	Beachwear & swimwear slogans	TYR Swimwear	Always in front.
Apparel slogans	Beachwear & swimwear slogans	Malibu Beachwear & Bikinis in Malaysia	Love the beach.
Apparel slogans	Beachwear & swimwear slogans	Jantzen, swimming suits and beach wear	All girls are gorgeous in Jantzen.
Apparel slogans	Beachwear & swimwear slogans	Jantzen, swimming suits and beach wear	Dive into life.
Apparel slogans	Beachwear & swimwear slogans	Jantzen, swimming suits and beach wear	Keep our beaches beautiful.
Apparel slogans	Beachwear & swimwear slogans	Miraclesuit Swimsuits	Look 10 lbs lighter in 10 seconds.
Apparel slogans	Beachwear & swimwear slogans	Dolphin Swimwear	Live in water.
Apparel slogans	Beachwear & swimwear slogans	Ujena Swimwear and Bikinis	Making women feel sexy since 1984.
Apparel slogans	Beachwear & swimwear slogans	Splashgear Swimwear	Now go get wet!
Apparel slogans	Beachwear & swimwear slogans	Zoggs, brand of goggles and swimwear	Share our passion for swimming!
Apparel slogans	Beachwear & swimwear slogans	Zoggs, brand of goggles and swimwear	Serious swimmers prefer Zoggs.
Apparel slogans	Beachwear & swimwear slogans	AussieBum, men's swimwear, surfwear and underwear	If you doubt yourself, wear something else!
Apparel slogans	Beachwear & swimwear slogans	Hive Swimwear Shop	Swimwear that sticks!
Apparel slogans	Beachwear & swimwear slogans	Wings Beachwear Stores in Florida	All you need to reach the beach!
Apparel slogans	Beachwear & swimwear slogans	Maru Swimwear	Dive into colour.
Apparel slogans	Beachwear & swimwear slogans	Nike Swimwear	Virtual skin.

Our input data is csv file consisting of 4 columns: *CATEGORY*, *SUBCATEGORY*, *BRAND*, *SLOGAN*. Data has been taken from:

http://www.textart.ru/database/slogan/list-advertising-slogans.html

CATEGORY column contains 21 different categories from various business branches and some of them are divided into more than one *SUBCATEGORY.*

In *BRAND* column we store name of company that is a creator of text specified in *SLOGAN* column. There are 3903 different brands and total of 5017 slogans inside our dataset.

Code description:

We start off with importing necessary libraries such as *sqlalchemy* and *pandas*. Next we define connection string and - using it - establish connection with database. Then we process of mapping existing tables to classes:

```
from sqlalchemy import create_engine,Column, Integer, String, Float, ForeignKey, Sequence, CheckConstraint, UniqueConstraint,
Date, Table, MetaData,select
from sqlalchemy.ext.declarative import declarative_base
import pandas as pd


db_string = "postgres://postgres:postgres@localhost:5432/slogans"


engine = create_engine(db_string)

Base = declarative_base()
```

Reading input data from file:

```
data = pd.read_csv('input_data.csv', delimiter=';')
data = data.drop('Unnamed: 4', 1)
data
```

Then we create list for each column containing unique values from that column:

```
all_category = data['CATEGORY'].unique()
print("category array: {0}".format(all_category))
```

```
all_subcategory = data['SUBCATEGORY'].unique()
print("subcategory array: {0}".format(all_subcategory))
```

```
all_brand = data['BRAND'].unique()
print("brand array: {0}".format(all_brand))
```

```
all_slogan = data['SLOGAN'].unique()
print("slogan array: {0}".format(all_slogan))
```

Creating SUBCATEGORY list related to ID of the corresponding CATEGORY:

```
subcategory_list = data[['CATEGORY','SUBCATEGORY']].drop_duplicates().reset_index().drop(columns = ['index']);
subcategory_list.index.name = 'id'

subcategory_list = subcategory_list.rename(columns = {'CATEGORY':'category_id'})
subcategory_list = subcategory_list.rename(columns = {'SUBCATEGORY':'subcategory'})

subcategory_list['category_id'] = subcategory_list['category_id'].map(lambda x:  category_list[category_list['category'] ==
x].index.values.astype(int)[0])

pd.set_option('display.max_rows', 100)
subcategory_list
```

We create 4 tables:

- *category* – containing all categories

- *subcategory* – containing all subcategories with relation to corresponding category.

- *brand* – containing all companies' names. There is no relation to subcategory because one company can own slogans from different subcategories.

- *slogan* – main table containing all slogan texts and relations do brands and subcategories.

Now we define classes representing tables in our database. Last line is responsible for creating complete table schema in database.

```
class category(Base):
    __tablename__ = 'category'
    __table_args__ = (
        UniqueConstraint('category'),
    )
    id = Column(Integer, Sequence('seq_category_id'), primary_key = True)
    category = Column(String(50))

class subcategory(Base):
    __tablename__ = 'subcategory'
    __table_args__ = (
        UniqueConstraint('subcategory'),
    )
    id = Column(Integer, Sequence('seq_subcategory_id'), primary_key=True)
    category_id=Column(Integer, ForeignKey('category.id'))
    subcategory = Column(String(100))

class brand(Base):
    __tablename__ = 'brand'
    __table_args__ = (
        UniqueConstraint('brand'),
    )
    id = Column(Integer, Sequence('seq_brand_id'), primary_key = True)
    brand = Column(String(200))

class slogan(Base):
    __tablename__ = 'slogan'
    id = Column(Integer,Sequence('seq_slogan_id'),  primary_key=True)
    slogan =Column(String(200))
    subcategory_id=Column(Integer, ForeignKey('subcategory.id'))
    brand_id=Column(Integer, ForeignKey('brand.id'))

Base.metadata.create_all(engine)
```

We add created lists to appropriate tables in database:

```
category_list.to_sql('category',engine, if_exists='append')
subcategory_list.to_sql('subcategory',engine, if_exists='append')
brand_list.to_sql('brand',engine, if_exists='append')
slogan_list.to_sql('slogan',engine, if_exists='append')
```

For validation purposes execute some *select* statements

- select all categories and count them:

```
mapper_stmt = select([dic_table['category'].columns.id,dic_table['category'].columns.category])
print('Mapper select: ')
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
print('\nIlosc kategorii:')
len(mapper_results)
```

```
Mapper select:
SELECT category.id, category.category
FROM category
[(0, 'Apparel slogans'), (1, 'Automotive slogans'), (2, 'Beauty slogans'), (3, 'Beverage slogans'), (4, 'Business slogans'),
(5, 'Construction slogans'), (6, 'Dining slogans'), (7, 'Educational slogans'), (8, 'Financial service slogans'), (9, 'Casino
slogans'), (10, 'Computers slogans'), (11, 'Condoms slogans'), (12, 'Magagines slogans'), (13, 'Motorcycle slogans'), (14, 'N
ewspapers slogans'), (15, 'Pet food slogans'), (16, 'Radio stations slogans'), (17, 'Real estate slogans'), (18, 'Tobacco slo
gans'), (19, 'Vitamins slogans'), (20, 'Watch slogans')]

Ilosc kategorii:
21
```

- selecting slogans from *Automotive slogans* category:

```
mapper_stmt = select([dic_table['slogan'].columns.slogan]).\
    where(dic_table['slogan'].columns.subcategory_id.in_(select([dic_table['subcategory'].columns.id]).\
        where(dic_table['subcategory'].columns.category_id.in_(select([dic_table['category'].columns.id]).\
            where(dic_table['category'].columns.category == 'Automotive slogans' )))))
print('Mapper select: ')
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
```

```
Mapper select:
SELECT slogan.slogan
FROM slogan
WHERE slogan.subcategory_id IN (SELECT subcategory.id
FROM subcategory
WHERE subcategory.category_id IN (SELECT category.id
FROM category
WHERE category.category = :category_1))
[('Blaupunkt. The advantage in your car.',), ('Kenwood. Listen to the Future.',), ('Rockford Fosgate. Car audio for fanatic
s.',), ('Volfenhag. A German Concept.',), ("Kicker. Fuel for Livin' Loud!",), ('Not just oil, Pennzoil.',), ('Havoline. Add M
ore Life to Your Car.',), ("Mobil 1. The oil that's changing oil.",), ('Unlike any other.',), ('Mercedes-Benz. The Future of
the Automobile.',), ('Engineered to move the human spirit.',), ('The Power of Dreams.',), ('It must be love.',), ('Honda. Fir
st man, then machine.',), ('Technology you can enjoy.',), ('The True Definition of Luxury. Yours.',), (' Acura. Precision Cra
fted Performance.',), ('Driven by passion. FIAT.',), ('Alfa Romeo. Beauty is not enough.',), ('Power for your control.',), ('
Volvo. For life.',), ('Subaru. Think. Feel. Drive.',), ("Driven By What's Inside.",), ('When You Get It, You Get It.',), ('Th
e Beauty of All-Wheel Drive.',), ('Ford. Feel the difference.',), ('  Ford. Bold moves.(USA)',), (' Built for life in Canad
a.(Canada)',), (' Built for the road ahead.',), ('  Ford. Designed for living. Engineered to last.',), ('  Have you driven a
Ford lately?',), ('Infiniti. Accelerating the Future.',), ('BMW. The Ultimate Driving Machine.',), ('BMW. Sheer Driving Pleas
```

- selecting 10 longest and shortest slogans:

```
# 10 shortest slogans
mapper_stmt = select([dic_table['slogan'].columns.id,dic_table['slogan'].columns.slogan]).\
    order_by(func.length(dic_table['slogan'].columns.slogan)).limit(10)
print('Mapper select: ')
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
```

```
Mapper select:
SELECT slogan.id, slogan.slogan
FROM slogan ORDER BY length(slogan.slogan)
 LIMIT :param_1
[(256, 'Shift'), (4622, 'Think.'), (1172, 'You in?'), (4353, 'Belong.'), (4252, 'Engage.'), (1782, 'Wake up.'),
(1687, 'Oh yeah.'), (1745, 'Be wild.'), (1232, 'Take me!'), (1746, 'No limit.')]
```

```
# 10  longest slogans
mapper_stmt = select([dic_table['slogan'].columns.id,dic_table['slogan'].columns.slogan]).\
    order_by(-func.length(dic_table['slogan'].columns.slogan,reverse='False')).limit(10)
print('Mapper select: ')
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
```

```
Mapper select:
SELECT slogan.id, slogan.slogan
FROM slogan ORDER BY -length(slogan.slogan)
 LIMIT :param_1
[(5010, 'The mainspring in a Bulova is made to last 256 years or 146 leather straps - whichever comes first.'),
(1097, 'A hard earned thirst needs a big cold beer and the best cold beer is Vic. Victoria Bitter.'), (1276, 'Was
hington state - the perfect climate for wine(The promotion of import Washington wine)'), (3531, 'Good times, good
friends, cold beer. Biltmore Bar and Grille, we're glad you're here.'), (4177, 'Western Libraries connects - peop
le to people, people to place, people to learning.'), (1076, 'Celebrate football(Note: Amstel - sponsor of the UE
FA Champions League 1994 - 2004)'), (3144, 'The internet is an ocean. Let me help you build your ship and navigat
e the waters.'), (1850, 'Specially made for healthy bodies, healthy lives, healthy kids(for Healthy Kids)'), (104
6, "No matter what what's-his-name says, I'm the prettiest and Lite's the greatest."), (4368, "There are some thi
ngs money can't buy. For everything else there's Mastercard.")]
```

We also define two additional functions and validate if functions work correctly.

- get_brand_slogans – returns all slogans owned by company that's name is passed as an argument.

```
#function finding all slogans of given brand
mapper_stmt = "CREATE OR REPLACE FUNCTION get_brand_slogan(brand_name VARCHAR)\
RETURNS TABLE (slogan VARCHAR) AS $$ \
BEGIN RETURN QUERY SELECT slogan.slogan FROM brand inner join slogan on brand.id=slogan.brand_id \
WHERE brand.brand ILIKE brand_name;\
END;$$ \
LANGUAGE 'plpgsql';"
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt)
```

```
CREATE OR REPLACE FUNCTION get_brand_slogan(brand_name VARCHAR)RETURNS TABLE (slogan VARCHAR) AS $$ BEGIN RETURN QU
ERY SELECT slogan.slogan FROM brand inner join slogan on brand.id=slogan.brand_id WHERE brand.brand ILIKE brand_nam
e;END;$$ LANGUAGE 'plpgsql';
```

```
mapper_stmt = "Select * from get_brand_slogan('Speedo Swimwear')"
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
```

```
Select * from get_brand_slogan('Speedo Swimwear')
[('Fueled by water.',), ('Speedo. Born in the water.',)]
```

```
mapper_stmt = "Select * from get_brand_slogan('Ground Round Grill & Bar')"
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
```

```
Select * from get_brand_slogan('Ground Round Grill & Bar')
[("We're even better than you remember.",), ("We've got what you like.",)]
```

-

get_subcategory_status – returns range in which number of slogans in subcategory passed as an argument is.

```
#function gettin the number of slogans in given subcategory
mapper_stmt ="""CREATE OR REPLACE FUNCTION get_subcategory_status(id_subcategory INTEGER)\
    RETURNS VARCHAR(25) AS $$\
DECLARE\
    total_slogans NUMERIC;\
    status VARCHAR(25);\
BEGIN\
    SELECT\
    INTO total_slogans count (slogan.subcategory_id)\
     FROM\
    subcategory  inner join slogan on subcategory.id=slogan.subcategory_id\
     WHERE\
    subcategory.id=id_subcategory;\
   CASE\
      WHEN total_slogans > 50 THEN\
         status = 'more than 50' ;\
      WHEN total_slogans >= 30 THEN\
          status = '30-50' ;\
      When total_slogans >= 10 THEN\
          status = '10-29' ;\
      ELSE\
          status = 'less than 10' ;\
   END CASE ;\
   RETURN status;\
END; $$ LANGUAGE 'plpgsql';"""
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt)
```

```
CREATE OR REPLACE FUNCTION get_subcategory_status(id_subcategory INTEGER)    RETURNS VARCHAR(25) AS $$DECLAR
E    total_slogans NUMERIC;    status VARCHAR(25);BEGIN    SELECT    INTO total_slogans count (slogan.subcat
egory_id)    FROM    subcategory  inner join slogan on subcategory.id=slogan.subcategory_id    WHERE    su
bcategory.id=id_subcategory;   CASE      WHEN total_slogans > 50 THEN         status = 'more than 50' ;
WHEN total_slogans >= 30 THEN          status = '30-50' ;      When total_slogans >= 10 THEN          status
= '10-29' ;      ELSE         status = 'less than 10' ;   END CASE ;   RETURN status;END; $$ LANGUAGE 'plpgs
ql';
```

```
mapper_stmt = "Select * from get_subcategory_status('34')"
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
```

```
Select * from get_subcategory_status('34')
[('30-50',)]
```

```
mapper_stmt = "Select * from get_subcategory_status('1')"
print(mapper_stmt)
mapper_results = engine.execute(mapper_stmt).fetchall()
print(mapper_results)
```

```
Select * from get_subcategory_status('1')
[('less than 10',)]
```