sql-programming

February 15, 2024

1 CS 1656 – Introduction to Data Science

1.1 Instructor: Alexandros Labrinidis

1.2 Teaching Assistants: Evangelos Karageorgos, Xiaoting Li, Zi Han Ding

1.2.1 Additional credits: Tahereh Arabghalizi, Zuha Agha, Anatoli Shein, Phuong Pham

1.3 ## Recitation : SQL via Data API

In this recitation, you will execute SQL queries on real data by connecting to the open data portal of Western Pennsylavnia Regional Data Center and requesting data via API calls.

[1]:

```
import json
from datetime import datetime, timedelta, date
import requests
import pandas as pd
import matplotlib.pyplot as plt

%matplotlib inline
```

We will be using Allegheny County Restaurant/Food Facility Inspection Violation Dataset found here https://data.wprdc.org/dataset/allegheny-county-restaurant-food-facility-inspection-violations. This dataset contains violation data from actual routine inspections by one of health department staff's members for the last two years. It should be fun to find out inspection results for places where we eat in Pittsburgh! =)

[2]:

```
wprdc_api_endpoint = "https://data.wprdc.org/api/3/action/datastore_search_sql"

# id for database table
resource_id = "1a1329e2-418c-4bd3-af2c-cc334e7559af"

# Get the date from 270 days ago)
# end_date = datetime.now()
# start_date = end_date - timedelta(days=270)

# Get two date endpoints
start_date = date(2021, 9, 1)
end_date = date(2022, 6, 1)
```

```python
# Convert to a string the format the the data center accepts (yyyy-mm-dd)
start_str = start_date.strftime("%Y-%m-%d")
end_str = end_date.strftime("%Y-%m-%d")

# SQL query we'll use in API call to request data
query = """
SELECT *
FROM "{}"
WHERE "inspect_dt" BETWEEN '{}' and '{}' AND "city" = '{}'""".
 ↪format(resource_id, start_str, end_str, "Pittsburgh")

# Make WPRDC API Call
response = requests.get(wprdc_api_endpoint, {'sql': query}, verify=False)

# Parse response JSON into python dictionary
response_data = json.loads(response.text)

# Convert dictionary to dataframe
df = pd.DataFrame.from_dict(response_data['result']['records'])

# Print the number of rows
print(df.shape[0], "rows total")
print(df.columns)
df.head()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\urllib3\connectionpool.py:1056:
InsecureRequestWarning: Unverified HTTPS request is being made to host
'data.wprdc.org'. Adding certificate verification is strongly advised. See:
https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(

10428 rows total
Index(['_id', '_full_text', 'encounter', 'id', 'placard_st', 'facility_name',
       'bus_st_date', 'description', 'description_new', 'num', 'street',
       'city', 'state', 'zip', 'inspect_dt', 'start_time', 'end_time',
       'municipal', 'rating', 'low', 'medium', 'high', 'url'],
      dtype='object')
```

```
[2]:         _id                                            _full_text     encounter  \
     0  87851551  '-01':24 '-04':8 '-11':23 '-111':32 '-15':9 '/…  202111010018
     1  87851552  '-01':26 '-04':8 '-11':25 '-111':34 '-15':9 '/…  202111010018
     2  87851553  '-01':21 '-04':8 '-11':20 '-111':29 '-15':9 '/…  202111010018
     3  87851784  '-03':21 '-06':7 '-109':29 '-11':20 '-15':8 '/…  202111030018
     4  87851785  '-03':21 '-06':7 '-109':29 '-11':20 '-15':8 '/…  202111030018

                  id placard_st         facility_name bus_st_date  \
     0  201004290004          1  Bryant Street Market  2010-04-15
```

```
1  201004290004          1  Bryant Street Market  2010-04-15
2  201004290004          1  Bryant Street Market  2010-04-15
3  201006070004          1        La Gourmandine  2010-06-15
4  201006070004          1        La Gourmandine  2010-06-15

                description  \
0  Retail/Convenience Store
1  Retail/Convenience Store
2  Retail/Convenience Store
3              Chain Bakery
4              Chain Bakery

                                     description_new   num  …    zip  \
0                  Certified Food Protection Manager  5901  …  15206
1  Contamination Prevention - Food, Utensils and …  5901  …  15206
2                                             Floors  5901  …  15206
3                                       Water Supply  4605  …  15201
4                             Handwashing Facilities  4605  …  15201

   inspect_dt start_time  end_time       municipal rating low medium  high  \
0  2021-11-01   12:30:00  13:45:00  Pittsburgh-111      V   F      T     F
1  2021-11-01   12:30:00  13:45:00  Pittsburgh-111      V   T      F     F
2  2021-11-01   12:30:00  13:45:00  Pittsburgh-111      V   T   None  None
3  2021-11-03   13:40:00  15:00:00  Pittsburgh-109      V   F      T     F
4  2021-11-03   13:40:00  15:00:00  Pittsburgh-109      V   T      T     F

                                                 url
0  http://appsrv.alleghenycounty.us/reports/rwser…
1  http://appsrv.alleghenycounty.us/reports/rwser…
2  http://appsrv.alleghenycounty.us/reports/rwser…
3  http://appsrv.alleghenycounty.us/reports/rwser…
4  http://appsrv.alleghenycounty.us/reports/rwser…

[5 rows x 23 columns]
```

Details of useful dataset attributes are below. ((Taken from https://data.wprdc.org/dataset/allegheny-county-restaurant-food-facility-inspection-violations/resource/1a1329e2-418c-4bd3-af2c-cc334e7559af)

- **facility_name**: the name of the facility
- **description**: Facility category
- **description_new**: The name of the potential violation
- **inspect_dt**: Date/time of the inspection
- **rating**: The result of the inspection ('V' for violation, other for non-violation)
- The health risk of a potential violation
- **low**: low risk
- **medium**: medium risk
- **high**: high risk

- The address of the facility
- **city**: The city
- **state**: The state
- **street**: The street
- **num**: The street number
- **zip**: The zip code

1.4 Queries

Q1) Find all unique decription categories of violation in Pittsburgh restaurants over the time span (violation description[violation]).

[3]:

```
query = """
SELECT DISTINCT "description_new" as violation
FROM "{}"
WHERE "inspect_dt" BETWEEN '{}' and '{}' AND "city" = '{}' """.
 ↪format(resource_id, start_str, end_str, "Pittsburgh")

response = requests.get(wprdc_api_endpoint, {'sql': query}, verify=False)

df = pd.DataFrame.from_dict(json.loads(response.text)['result']['records'])

df
```

```
C:\ProgramData\Anaconda3\lib\site-packages\urllib3\connectionpool.py:1056:
InsecureRequestWarning: Unverified HTTPS request is being made to host
'data.wprdc.org'. Adding certificate verification is strongly advised. See:
https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
```

[3]:

```
                                             violation
0                                       Administrative
1                     Certified Food Protection Manager
2                             Cleaning and Sanitization
3                           Cold Holding Temperatures
4                                     Consumer Advisory
5    Contamination Prevention - Food, Utensils and …
6                                Cooking Temperatures
7                                         Cooling Food
8                            Cross-Contamination Prevention
9                                  Date Marking of Food
10                             Demonstration of Knowledge
11                         Dressing rooms and Locker rooms
12                             Employee Personal Hygiene
13   Fabrication, Design, Installation and Maintenance
14                        Facilities to Maintain Temperature
15                                               Floors
16                                Food Source/Condition
```

```
17                                 Garbage and Refuse
18                                   General Premises
19                             Handwashing Facilities
20                           Hot Holding Temperatures
21                                           Lighting
22                                    Pest Management
23                                           Plumbing
24                            Probe-Type Thermometers
25                             Reheating Temperatures
26                                        Toilet Room
27                                        Toxic Items
28                                        Ventilation
29                                 Walls and ceilings
30                               Waste Water Disposal
31                                       Water Supply
```

Q2) Find restaurants in Pittsburgh with no violations in at least one decription category (facility name[facility], number of violations[count]). NOTE: a facility has a violation if the inspection rating has the value 'V'.

[4]:

```
query = """
SELECT "facility_name" as facility, COUNT("description_new") as count
FROM "{}"
WHERE "inspect_dt" BETWEEN '{}' and '{}' AND "city" = '{}' AND "rating" <> '{}'
GROUP BY "facility_name" """.format(resource_id, start_str, end_str,␣
 ↪"Pittsburgh", "V")

response = requests.get(wprdc_api_endpoint, {'sql': query}, verify=False)

df = pd.DataFrame.from_dict(json.loads(response.text)['result']['records'])

df
```

```
C:\ProgramData\Anaconda3\lib\site-packages\urllib3\connectionpool.py:1056:
InsecureRequestWarning: Unverified HTTPS request is being made to host
'data.wprdc.org'. Adding certificate verification is strongly advised. See:
https://urllib3.readthedocs.io/en/1.26.x/advanced-usage.html#ssl-warnings
  warnings.warn(
```

[4]:

```
                                           facility  count
0                                2O2 Hometown Tacos      1
1                                  Aladdin's Eatery      1
2            All of Us Care / Volunteers of America      1
3                                               Bao      1
4                         Bar Marco @ the Firehouse      1
..                                              ...    ...
69                           UPMC Mercy / 1847 Cafe      1
```

```
70        Vickey's Soul Grill Restaurant & Catering      1
71                             Victory Banquet Hall      1
72                                     Vocelli Pizza      2
73  Yeshiva Nechama Minsky Girls School & Preschool      1

[74 rows x 2 columns]
```

2 Tasks

Tasks 1 to 4 must be implemented in Task.py.

For all tasks, we want the results in Pittsburgh, over the specified time span, and all queries are about violations (rating is ‘V’).

As the API returns the results as a list of dictionaries, the order of the query columns is irrelevant.

T1) Find the top 20 facilities that start with ‘Pitt’ and have the highest counts of violations *(facility name[facility], number of violations[count])*.

[ ]:

T2) Find the top 18 restaurants with the maximum number of violations *(facility name[facility], number of violations[count])*. Include all results in case of a tie (For example, if the 18th top restaurant has 10 violations, incude all other restaurants with 10 violations). HINT: You will need an extra query to get the tie-breaker value.

[ ]:

T3) Find the facilities that start with ‘Pitt’ and have violations over the time span *(violation description[violation], number of facilities[count], facility names[facilities])*. The *facilities* field must be a concatenation of all facility names, in alphabetical order, seperated by a comma and a space (‘,’).

[ ]:

Now lets look at all facilities that contain word ‘Pitt’.

T4) Find the category descriptions and their high, medium, low risk ratings for all violations at all facilities that have word ‘Pitt’ in their name. Note that results that contain word ‘Pitt’ as part of another word (e.g. ‘Pittsburgh’) should not be included *(facility name[facility], violation description[violation], high[high], medium[medium], low[low])*. HINT: consider *all* edge cases for identifying ‘Pitt’ as a seperate word.

[ ]: