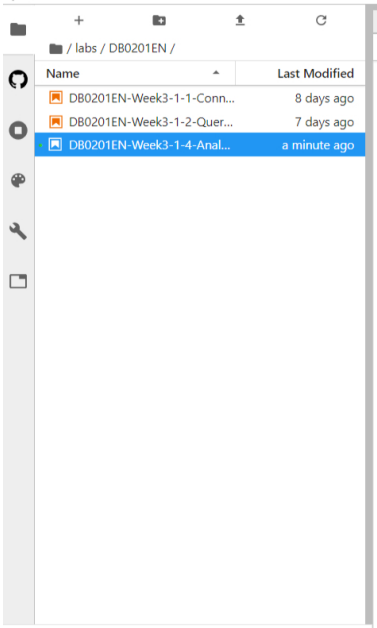


Launcher | DB0201EN-Week3-1-4-Anal

Code | Python

Lab: Analyzing a real world data-set with SQL and Python

Introduction

This notebook shows how to store a dataset into a database using and analyze data using SQL and Python. In this lab you will:

1. Understand a dataset of selected socioeconomic indicators in Chicago
2. Learn how to store data in an Db2 database on IBM Cloud instance
3. Solve example problems to practice your SQL skills

Selected Socioeconomic Indicators in Chicago

The city of Chicago released a dataset of socioeconomic data to the Chicago City Portal. This dataset contains a selection of six socioeconomic indicators of public health significance and a "hardship index," for each Chicago community area, for the years 2008 – 2012.

Scores on the hardship index can range from 1 to 100, with a higher index number representing a greater level of hardship.

A detailed description of the dataset can be found on the city of Chicago's website, but to summarize, the dataset has the following variables:

- **Community Area Number** (`ca`): Used to uniquely identify each row of the dataset
- **Community Area Name** (`community_area_name`): The name of the region in the city of Chicago
- **Percent of Housing Crowded** (`percent_of_housing_crowded`): Percent of occupied housing units with more than one person per room
- **Percent Households Below Poverty** (`percent_households_below_poverty`): Percent of households living below the federal poverty line
- **Percent Aged 16+ Unemployed** (`percent_aged_16_unemployed`): Percent of persons over the age of 16 years that are unemployed
- **Percent Aged 25+ without High School Diploma** (`percent_aged_25_without_high_school_diploma`): Percent of persons over the age of 25 years without a high school education
- **Percent Aged Under** 18 or Over 64:Percent of population under 18 or over 64 years of age (`percent_aged_under_18_or_over_64`): (ie. dependents)
- **Per Capita Income** (`per_capita_income_`): Community Area per capita income is estimated as the sum of tract-level aggragate incomes divided by the total population
- **Hardship Index** (`hardship_index`): Score that incorporates each of the six selected socioeconomic indicators

In this Lab, we'll take a look at the variables in the socioeconomic indicators dataset and do some basic analysis with Python.

Connect to the database

Let us first load the SQL extension and establish a connection with the database

[1]:
```
%load_ext sql
```

[2]:
```
# Remember the connection string is of the format:
# %sql ibm_db_sa://my-username:my-password@my-hostname:my-port/my-db-name
# Enter the connection string for your Db2 on Cloud database instance below
# i.e. copy after db2:// from the URI string in Service Credentials of your Db2 instance. Remove the double quotes at the end.
%sql ibm_db_sa://gkm89241:w13fqqz1cdc%5Epqqr@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
```

[2]:
```
'Connected: gkm89241@BLUDB'
```

Store the dataset in a Table

In many cases the dataset to be analyzed is available as a .CSV (comma separated values) file, perhaps on the internet. To analyze the data using SQL, it first needs to be stored in the database.

We will first read the dataset source .CSV from the internet into pandas dataframe

Then we need to create a table in our Db2 database to store the dataset. The PERSIST command in SQL "magic" simplifies the process of table creation and writing the data from a `pandas` dataframe into the table

[3]:
```
import pandas
chicago_socioeconomic_data = pandas.read_csv('https://data.cityofchicago.org/resource/jcxq-k9xf.csv')
%sql PERSIST chicago_socioeconomic_data
```

```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
```

[3]:
```
'Persisted chicago_socioeconomic_data'
```

You can verify that the table creation was successful by making a basic query like:

[4]:
```
%sql SELECT * FROM chicago_socioeconomic_data limit 5;
```

```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
Done.
```

[4]:

percent_households_below_poverty	percent_aged_16_unemployed	percent_aged_25_without_high_school_diploma	percent_aged_under_18_or_over_64	per_capita_income_	hardship_index
23.6	8.7	18.2	27.5	23939	39.0
17.2	8.8	20.8	38.5	23040	46.0
24.0	8.9	11.8	22.2	35787	20.0
10.9	8.2	13.4	25.5	37524	17.0

7.5	5.2	4.5	26.2	57123	6.0

Problems

Problem 1

How many rows are in the dataset?

[5]:
```
5
```

[5]:
```
5
```

Double-click **here** for the solution.

Problem 2

How many community areas in Chicago have a hardship index greater than 50.0?

[6]:
```
0
```

[6]:
```
0
```

Double-click **here** for the solution.

Problem 3

What is the maximum value of hardship index in this dataset?

Did you know? IBM Watson Studio lets you build and deploy an AI solution, using the best of open source and IBM software and giving your team a single environment to work in. Learn more here.

[7]:
```
%sql SELECT MAX (hardship_index) FROM chicago_socioeconomic_data;
```

```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
Done.
```

[7]:

1
98.0

```
Double-click __here__ for the solution.

<!-- Hint:

%sql SELECT MAX(hardship_index) FROM chicago_socioeconomic_data;

Correct answer: 98.0
-->
```

Problem 4

Which community area which has the highest hardship index?

[20]:
```
%sql select community_area_name from chicago_socioeconomic_data where hardship_index = ( select max(hardship_index) from chicago_socioeconomic_data )
```

```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
Done.
```

[20]:

community_area_name
Riverdale

```
Double-click __here__ for the solution.

<!-- Hint:

## We can use the result of the last query to as an input to this query:
%sql SELECT community_area_name FROM chicago_socioeconomic_data where hardship_index=98.0

## or another option:
%sql SELECT community_area_name FROM chicago_socioeconomic_data ORDER BY hardship_index DESC NULLS LAST FETCH FIRST ROW ONLY;

## or you can use a sub-query to determine the max hardship index:
%sql select community_area_name from chicago_socioeconomic_data where hardship_index = ( select max(hardship_index) from chicago_socioeconomic_data )

Correct answer: 'Riverdale'
-->
```

Problem 5

Which Chicago community areas have per-capita incomes greater than $60,000?

[22]:
```
%sql select community_area_name from chicago_socioeconomic_data where per_capita_income_ > 60000
```

```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
Done.
```

[22]:

community_area_name
Lake View
Lincoln Park
Near North Side
Loop

```
Double-click __here__ for the solution.

<!-- Hint:

%sql SELECT community_area_name FROM chicago_socioeconomic_data WHERE per_capita_income_ > 60000;

Correct answer:Lake View,Lincoln Park, Near North Side, Loop
-->
```

Problem 6

Create a scatter plot using the variables `per_capita_income_` and `hardship_index`. Explain the correlation between the two variables.

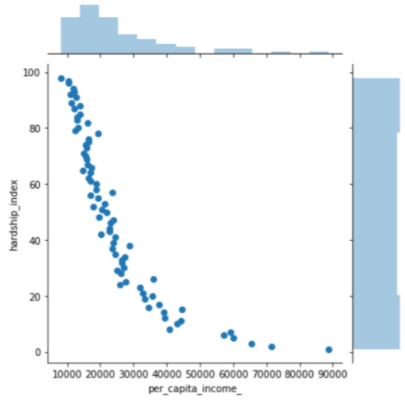
[23]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

[24]:
```
income_vs_hardship = %sql SELECT per_capita_income_, hardship_index FROM chicago_socioeconomic_data;
```

```
plot = sns.jointplot(x='per_capita_income_',y='hardship_index', data=income_vs_hardship.DataFrame())
```

```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
Done.
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/scipy/stats/stats.py:1713: FutureWarning: Using a non-tuple sequence for multidimensional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted as an array index, `arr[np.array(seq)]`, which will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

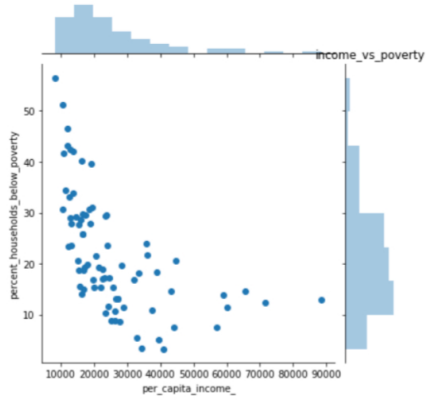


•••

[27]:
```
income_vs_poverty = %sql SELECT per_capita_income_, percent_households_below_poverty FROM chicago_socioeconomic_data;
plot = sns.jointplot(x='per_capita_income_',y='percent_households_below_poverty', data=income_vs_poverty.DataFrame())
plt.title('income_vs_poverty')
```

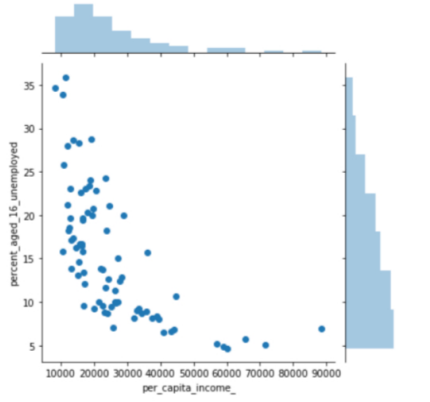
```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
Done.
```

[27]:
```
Text(0.5, 1, 'income_vs_poverty')
```



[28]:
```
income_vs_unemployed = %sql SELECT per_capita_income_, percent_aged_16_unemployed FROM chicago_socioeconomic_data;
plot = sns.jointplot(x='per_capita_income_',y='percent_aged_16_unemployed', data=income_vs_unemployed.DataFrame())
```

```
 * ibm_db_sa://gkm89241:***@dashdb-txn-sbox-yp-lon02-01.services.eu-gb.bluemix.net:50000/BLUDB
Done.
```



[ ]:

[ ]:

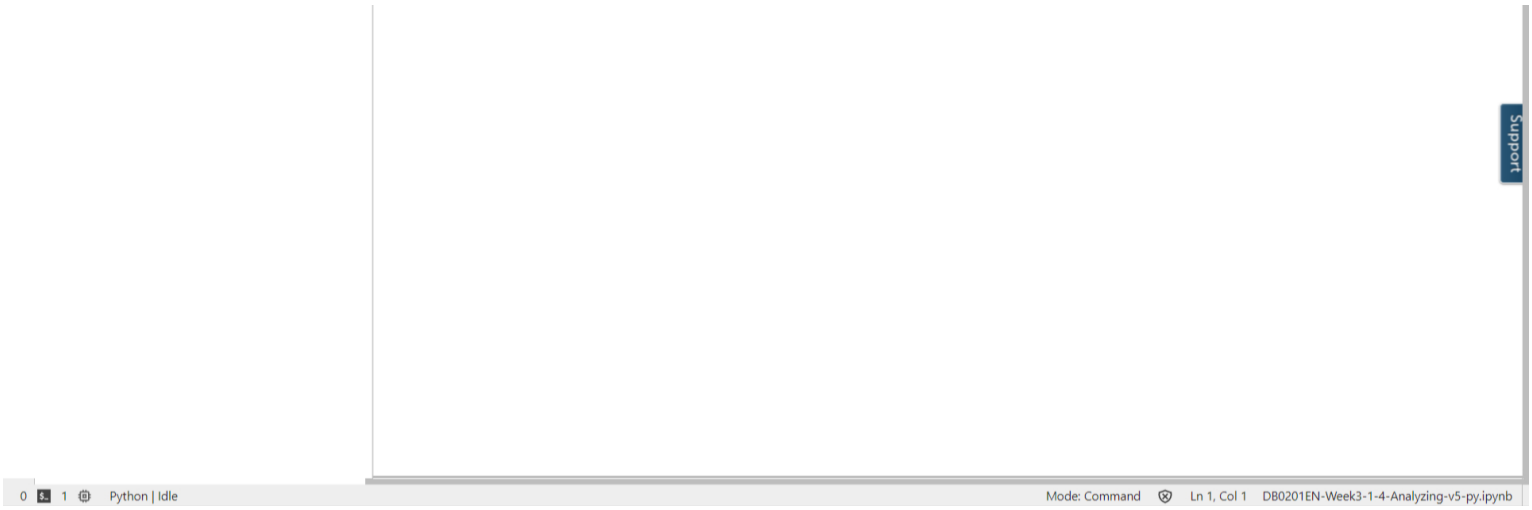
[ ]:

Conclusion

Now that you know how to do basic exploratory data analysis using SQL and python visualization tools, you can further explore this dataset to see how the variable `per_capita_income_` is related to `percent_households_below_poverty` and `percent_aged_16_unemployed`. Try to create interesting visualizations!

Summary

In this lab you learned how to store a real world data set from the internet in a database (Db2 on IBM Cloud), gain insights into data using SQL queries. You also visualized a portion of the data in the database to see what story it tells.


Support
0
1
Python | Idle
Mode: Command
Ln 1, Col 1
DB0201EN-Week3-1-4-Analyzing-v5-py.ipynb