**Use Python Pandas (Python Data Analysis Library) on the "Titanic Dataset" with a direct SQL statements comparison**

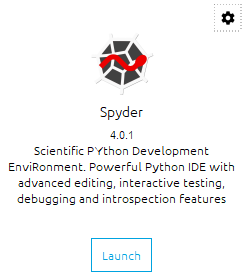
*Python code by Neven Dujmović*

Few words about the tool that I use for my python coding:

My recommendation is to use **Anaconda** distribution for the Python for data analysis and other purposes. This great tool can be downloaded on:

<https://www.anaconda.com/products/individual>

Use **Anaconda Individual Edition**, and you can install it with default features. After it is installed, launch **Spyder** Python IDE in **Anaconda Navigator** to run the data extraction & analytics queries.



On the first line of the code, you have to import "**pandas**" library that holds all you will need for data analytics.

|  |
| --- |
| import pandas as pd |

On the next line, we will load the data from the comma-separated (CSV) text file that contains a "titanic" dataset. The file must be on the same location on the file system as your python script (.py), or you will have to specify the full path to the .csv file.

|  |
| --- |
| titanic\_passengers = pd.read\_csv('titanic\_passengers\_data\_sample.csv') |

Now, we can try to do some queries with Python Pandas, and this will be done in the following way: 😊

* for standard SQL statement used for data extraction or data analytics,
* a corresponding Python Pandas code will be provided that will do the same data extraction or data analytics.

1. **Limit resulting rows in the simple query**

|  |  |
| --- | --- |
| **SQL** | SELECT \* FROM titanic\_passengers LIMIT 5 |

|  |  |
| --- | --- |
| **Python** | query = titanic\_passengers.head() |

|  |  |
| --- | --- |
| **SQL** | SELECT \* FROM titanic\_passengers LIMIT 20 |

|  |  |
| --- | --- |
| **Python** | query01 = titanic\_passengers.head(20) |

1. **Select specific columns**

|  |  |
| --- | --- |
| **SQL** | SELECT Name, Pclass, Survived FROM titanic\_passengers |

|  |  |
| --- | --- |
| **Python** | query02 = titanic\_passengers[['Name', 'Pclass', 'Survived']] |

|  |  |
| --- | --- |
| **SQL** | SELECT PassengerId, Name, Age FROM titanic\_passengers |

|  |  |
| --- | --- |
| **Python** | query03 = titanic\_passengers[['PassengerId', 'Name', 'Age']] |

1. **Return unique rows from the specific column**

|  |  |
| --- | --- |
| **SQL** | SELECT DISTINCT Pclass FROM titanic\_passengers |

|  |  |
| --- | --- |
| **Python** | query04 = titanic\_passengers.drop\_duplicates('Pclass')[['Pclass']] |

1. **Extract only those records that fulfill a defined criterion**

|  |  |
| --- | --- |
| **SQL** | SELECT Name, Sex, Age, Pclass, Fare  FROM titanic\_passengers WHERE Fare<8 |

|  |  |
| --- | --- |
| **Python** | query05 = titanic\_passengers[['Name', 'Sex', 'Age', 'Pclass', 'Fare']]\  .loc[titanic\_passengers['Fare'] < 8] |

**Note:** "\" is used to go continue python line of code to a next line

|  |  |
| --- | --- |
| **SQL** | SELECT Name, Age, Pclass, Fare, Embarked  FROM titanic\_passengers  WHERE Fare>250 AND (Embarked = 'S' OR Embarked = 'C') |

|  |  |
| --- | --- |
| **Python** | query06 = titanic\_passengers[['Name', 'Age', 'Pclass', 'Fare', 'Embarked']]\  .loc[(titanic\_passengers['Fare'] > 250) & \  ((titanic\_passengers['Embarked'] == 'S') | \  (titanic\_passengers['Embarked'] == 'C'))] |

1. **extract only those records that fulfill a criterion that contains a part of a substring in the specific column**

|  |  |
| --- | --- |
| **SQL** | SELECT Name, Age, Pclass, SibSp, Parch  FROM titanic\_passengers  WHERE Name like '%Fortune%' |

|  |  |
| --- | --- |
| **Python** | query07 = titanic\_passengers[['Name', 'Age', 'Pclass', 'SibSp', 'Parch']]\  .loc[titanic\_passengers['Name'].str.contains('Fortune')] |

1. **Get an aggregate result as a single value that is calculated from the values in a specific column**

|  |  |
| --- | --- |
| **SQL** | SELECT MAX(Fare) FROM titanic\_passengers |

|  |  |
| --- | --- |
| **Python** | query08 = titanic\_passengers['Fare'].max() |

|  |  |
| --- | --- |
| **SQL** | SELECT COUNT(PassengerId) AS survived\_passengers  FROM titanic\_passengers  WHERE Survived = 'Yes' |

|  |  |
| --- | --- |
| **Python** | query09 = titanic\_passengers[['PassengerId']].\  loc[(titanic\_passengers['Survived'] == 'Yes')].count(axis = 0).\  rename(index={'PassengerId': 'survived\_passengers'}) |

**Note:** axis = 0 indicates row

|  |  |
| --- | --- |
| **SQL** | SELECT Pclass as passenger\_class, AVG(Fare) as average\_price  FROM titanic\_passengers  GROUP BY Pclass |

|  |  |
| --- | --- |
| **Python** | query10 = titanic\_passengers.groupby('Pclass')['Fare'].mean()\  .reset\_index()\  .rename(columns={'Pclass': 'passenger\_class', 'Fare': 'average\_price'}) |

1. **Extract sorted data with a limited number of rows in the results**

|  |  |
| --- | --- |
| **SQL** | SELECT Survived, Name, Age, Sex FROM titanic\_passengers  WHERE Survived = 'Yes'  ORDER BY Age DESC  LIMIT 10 |

|  |  |
| --- | --- |
| **Python** | query11 = titanic\_passengers[['Survived', 'Name', 'Age', 'Sex']]\  .loc[(titanic\_passengers['Survived'] == 'Yes')]\  .sort\_values(by=['Age'], ascending=False).head(10) |

1. **Test for the empty values**

|  |  |
| --- | --- |
| **SQL** | SELECT COUNT(PassengerId) AS no\_age\_passengers  FROM titanic\_passengers  WHERE Age IS NULL |

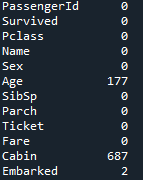
|  |  |
| --- | --- |
| **Python** | query12 = len(titanic\_passengers) - titanic\_passengers['Age'].count() |

or

|  |  |
| --- | --- |
| **Python** | query12a = sum(pd.isnull(titanic\_passengers['Age'])) |

1. **Identify the columns with missing values along with the count and print it on the console**

|  |  |
| --- | --- |
| **Python** | print (titanic\_passengers.isnull().sum(axis=0)) |



**Queries for a practice:**

1. **Calculate the number of passengers embarked from the three different ports: (C=Cherburg, S=Southampton, Q=Queenstown)**

|  |  |
| --- | --- |
| **SQL** | SELECT Embarked, COUNT(\*) FROM titanic\_passengers  WHERE (Embarked = 'C' OR Embarked = 'S' OR Embarked = 'Q')  GROUP BY Embarked  ORDER BY COUNT(\*) DESC; |

|  |  |
| --- | --- |
| **Python** | practice\_query01 = titanic\_passengers[['Embarked', 'PassengerId']]\  .loc[(titanic\_passengers['Embarked'] == 'C') | \  (titanic\_passengers['Embarked'] == 'S') | \  (titanic\_passengers['Embarked'] == 'Q')]    practice\_query01 = practice\_query01.groupby('Embarked')['PassengerId']\  .count()\  .reset\_index(name='count')\  .sort\_values(['count'], ascending=False) |

1. **What was the average age of men & women, respectively, in each Passenger Class?**

|  |  |
| --- | --- |
| **SQL** | SELECT Pclass, Sex, AVG(Age) FROM titanic\_passengers  GROUP BY Pclass, Sex  ORDER BY Pclass ASC; |

|  |  |
| --- | --- |
| **Python** | practice\_query02 = titanic\_passengers[['Pclass', 'Sex', 'Age', 'PassengerId']]\  .groupby(['Pclass', 'Sex'])['Age']\  .mean()\  .reset\_index(name='Average age')\  .sort\_values(['Pclass'], ascending=True) |

1. **Who are the top 20 passengers who paid the most expensive Fares?**

|  |  |
| --- | --- |
| **SQL** | SELECT Name, Fare FROM titanic\_passengers  ORDER BY Fare DESC  LIMIT 20; |

|  |  |
| --- | --- |
| **Python** | practice\_query03 = titanic\_passengers[['Name', 'Fare']]\  .sort\_values(['Fare'], ascending=False).head(20) |

1. **What was the average age of 3rd class non survived men?**

|  |  |
| --- | --- |
| **SQL** | SELECT AVG(Age) FROM titanic\_passengers  WHERE Sex = 'male' AND Survived = 'No' AND Pclass = 3; |

|  |  |
| --- | --- |
| **Python** | practice\_query04 = \  titanic\_passengers[['Pclass', 'Sex', 'Age', 'Survived', 'PassengerId']]\  .loc[(titanic\_passengers['Sex'] == 'male') & \  (titanic\_passengers['Survived'] == 'No') & \  (titanic\_passengers['Pclass'] == 3)]  practice\_query04 = practice\_query04[['Age', 'PassengerId']]['Age'].mean() |

1. **What was the average age of 1st class survived women?**

|  |  |
| --- | --- |
| **SQL** | SELECT AVG(Age) FROM titanic\_passengers  WHERE Sex = 'female' AND Survived = 'Yes' AND Pclass = 1; |

|  |  |
| --- | --- |
| **Python** | practice\_query05 = \  titanic\_passengers[['Pclass', 'Sex', 'Age', 'Survived', 'PassengerId']]\  .loc[(titanic\_passengers['Sex'] == 'female') & \  (titanic\_passengers['Survived'] == 'Yes') & \  (titanic\_passengers['Pclass'] == 1)]  practice\_query05 = practice\_query05[['Age', 'PassengerId']]['Age'].mean() |