Data Science amb Python

Sprint 18

S18 T01: NoSQL database task

Cristiane de Souza da Silva

July 2021

Description

We are starting to get acquainted with NoSQL databases !!! Let's start with a few basic exercises

Nivel 1

Exercise 1

Create a NoSQL database using MongoDB. Add some sample data that allows you to check that you are able to process the information in a basic way.

Exercise 2

Connect the NoSQL database to Python using for example pymongo.

In [1]:
```
! pip install pymongo
```

```
Requirement already satisfied: pymongo in /Applications/anaconda3/lib/pytho
n3.8/site-packages (3.11.4)
```

In [7]:
```
# import libraies
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import pymongo
```

In [8]:
```
# Creating a Database

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["mydatabase"]
mydb
```

Out[8]:
```
Database(MongoClient(host=['localhost:27017'], document_class=dict, tz_awar
e=False, connect=True), 'mydatabase')
```

In [9]:
```
#Check if Database Exists

print(myclient.list_database_names())
```

```
['admin', 'config', 'local']
```

In [ ]:
```

```

In [41]:
```
df_teste = pd.read_csv('test_scores.csv')
df_teste.values.tolist()
```

Out[41]:
```
[['ANKYI',
  'Urban',
  'Non-public',
  '6OL',
  'Standard',
  20.0,
  '2FHT3',
  'Female',
  'Does not qualify',
  62.0,
  72.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  '6OL',
  'Standard',
  20.0,
  '3JIVH',
  'Female',
  'Does not qualify',
  66.0,
  79.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  '6OL',
  'Standard',
  20.0,
  '3XOWE',
  'Male',
  'Does not qualify',
  64.0,
  76.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  '6OL',
  'Standard',
  20.0,
  '556O0',
  'Female',
  'Does not qualify',
  61.0,
  77.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  '6OL',
```

```
 'Standard',
 20.0,
 '74LOE',
 'Male',
 'Does not qualify',
 64.0,
 76.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 '7YZO8',
 'Female',
 'Does not qualify',
 66.0,
 74.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 '9KMZD',
 'Male',
 'Does not qualify',
 63.0,
 75.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 '9USQK',
 'Female',
 'Does not qualify',
 63.0,
 72.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'CS5QP',
 'Male',
 'Does not qualify',
 64.0,
 77.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'D6HT8',
 'Female',
 'Does not qualify',
 61.0,
```

```
 72.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'DZMKU',
 'Male',
 'Does not qualify',
 61.0,
 73.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'FH7B9',
 'Male',
 'Does not qualify',
 64.0,
 74.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'JI9VG',
 'Male',
 'Does not qualify',
 66.0,
 78.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'JQM2W',
 'Female',
 'Does not qualify',
 60.0,
 71.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'MEUC4',
 'Female',
 'Does not qualify',
 64.0,
 77.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
```

```
 20.0,
 'R4U8H',
 'Male',
 'Does not qualify',
 64.0,
 73.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'TH7KI',
 'Male',
 'Does not qualify',
 63.0,
 70.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'U1FV7',
 'Male',
 'Does not qualify',
 67.0,
 73.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'WC5I6',
 'Female',
 'Does not qualify',
 63.0,
 71.0],
['ANKYI',
 'Urban',
 'Non-public',
 '6OL',
 'Standard',
 20.0,
 'ZBQ4T',
 'Male',
 'Does not qualify',
 64.0,
 73.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 '0CRO6',
 'Male',
 'Does not qualify',
 60.0,
 68.0],
```

```
 ['ANKYI',
  'Urban',
  'Non-public',
  'ZNS',
  'Standard',
  21.0,
  '1QMDI',
  'Male',
  'Does not qualify',
  60.0,
  70.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  'ZNS',
  'Standard',
  21.0,
  '3CFUK',
  'Male',
  'Does not qualify',
  57.0,
  66.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  'ZNS',
  'Standard',
  21.0,
  '447O0',
  'Male',
  'Does not qualify',
  57.0,
  70.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  'ZNS',
  'Standard',
  21.0,
  '4IDEM',
  'Female',
  'Qualifies for reduced/free lunch',
  56.0,
  65.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  'ZNS',
  'Standard',
  21.0,
  '5UQNP',
  'Male',
  'Does not qualify',
  58.0,
  67.0],
 ['ANKYI',
  'Urban',
  'Non-public',
  'ZNS',
  'Standard',
  21.0,
```

```
 'AV95M',
 'Female',
 'Does not qualify',
 60.0,
 70.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'BX6I6',
 'Female',
 'Does not qualify',
 60.0,
 67.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'EYFXR',
 'Male',
 'Does not qualify',
 54.0,
 63.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'FFC9M',
 'Male',
 'Does not qualify',
 60.0,
 75.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'GGT4A',
 'Male',
 'Does not qualify',
 58.0,
 66.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'HRZOJ',
 'Female',
 'Does not qualify',
 66.0,
 71.0],
['ANKYI',
```

```
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'IFC62',
 'Male',
 'Does not qualify',
 60.0,
 76.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'LLQI3',
 'Female',
 'Does not qualify',
 59.0,
 69.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'OCJE3',
 'Female',
 'Does not qualify',
 57.0,
 69.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'OGW1F',
 'Female',
 'Does not qualify',
 60.0,
 68.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'Q5QRY',
 'Female',
 'Does not qualify',
 61.0,
 71.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'TJRIV',
```

```
 'Female',
 'Does not qualify',
 61.0,
 66.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'U24U5',
 'Female',
 'Does not qualify',
 59.0,
 69.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'VKX2N',
 'Female',
 'Does not qualify',
 62.0,
 72.0],
['ANKYI',
 'Urban',
 'Non-public',
 'ZNS',
 'Standard',
 21.0,
 'ZTROF',
 'Male',
 'Does not qualify',
 60.0,
 66.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 '1IALS',
 'Female',
 'Does not qualify',
 61.0,
 75.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 '5NDXD',
 'Male',
 'Qualifies for reduced/free lunch',
 58.0,
 78.0],
['CCAAW',
 'Suburban',
```

```
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 '6DCTV',
 'Female',
 'Qualifies for reduced/free lunch',
 64.0,
 82.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 '6OB0S',
 'Male',
 'Qualifies for reduced/free lunch',
 58.0,
 77.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'AITPY',
 'Male',
 'Does not qualify',
 65.0,
 87.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'B8B6G',
 'Male',
 'Does not qualify',
 65.0,
 80.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'C0UW3',
 'Female',
 'Qualifies for reduced/free lunch',
 62.0,
 75.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'CD1MB',
 'Male',
```

```
 'Qualifies for reduced/free lunch',
 58.0,
 73.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'F4ZHY',
 'Male',
 'Qualifies for reduced/free lunch',
 59.0,
 74.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'GT9F2',
 'Male',
 'Qualifies for reduced/free lunch',
 59.0,
 74.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'L5T1W',
 'Male',
 'Qualifies for reduced/free lunch',
 63.0,
 76.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'MVXN3',
 'Female',
 'Does not qualify',
 63.0,
 78.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'N7XOT',
 'Male',
 'Does not qualify',
 59.0,
 77.0],
['CCAAW',
 'Suburban',
 'Non-public',
```

```
 '2B1',
 'Experimental',
 18.0,
 'NKP72',
 'Male',
 'Qualifies for reduced/free lunch',
 61.0,
 77.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'QJVAM',
 'Male',
 'Qualifies for reduced/free lunch',
 55.0,
 72.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'V6ZA8',
 'Male',
 'Does not qualify',
 66.0,
 83.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'X4NP9',
 'Female',
 'Does not qualify',
 67.0,
 81.0],
['CCAAW',
 'Suburban',
 'Non-public',
 '2B1',
 'Experimental',
 18.0,
 'YEJFP',
 'Female',
 'Does not qualify',
 68.0,
 79.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '2RA1H',
 'Female',
 'Does not qualify',
```

```
 63.0,
 80.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '2U37U',
 'Female',
 'Qualifies for reduced/free lunch',
 63.0,
 83.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '30219',
 'Male',
 'Does not qualify',
 66.0,
 82.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '3RJO7',
 'Female',
 'Does not qualify',
 67.0,
 84.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '42OS6',
 'Female',
 'Qualifies for reduced/free lunch',
 67.0,
 83.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '4B72M',
 'Male',
 'Does not qualify',
 73.0,
 85.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
```

```
 'Experimental',
 20.0,
 '7D9JO',
 'Male',
 'Does not qualify',
 68.0,
 87.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '7QNXG',
 'Female',
 'Qualifies for reduced/free lunch',
 70.0,
 91.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '8HMCH',
 'Female',
 'Qualifies for reduced/free lunch',
 74.0,
 85.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 '8W6N8',
 'Female',
 'Does not qualify',
 64.0,
 83.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'BVGDA',
 'Male',
 'Qualifies for reduced/free lunch',
 62.0,
 78.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'CFICH',
 'Female',
 'Does not qualify',
 76.0,
```

```
 84.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'DTGZA',
 'Female',
 'Does not qualify',
 73.0,
 86.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'EEJ6S',
 'Male',
 'Does not qualify',
 65.0,
 87.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'JSYRI',
 'Female',
 'Does not qualify',
 66.0,
 79.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'KGFXN',
 'Female',
 'Does not qualify',
 66.0,
 84.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'KTSGX',
 'Female',
 'Does not qualify',
 66.0,
 82.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
```

```
 20.0,
 'SAKZI',
 'Female',
 'Qualifies for reduced/free lunch',
 65.0,
 83.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'XAHWJ',
 'Female',
 'Qualifies for reduced/free lunch',
 63.0,
 78.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'EPS',
 'Experimental',
 20.0,
 'XP6HQ',
 'Female',
 'Qualifies for reduced/free lunch',
 68.0,
 84.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 '0ZAWS',
 'Female',
 'Does not qualify',
 65.0,
 79.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 '4CE6S',
 'Female',
 'Does not qualify',
 70.0,
 83.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'DB15U',
 'Female',
 'Qualifies for reduced/free lunch',
 65.0,
 81.0],
```

```
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'GFIP5',
 'Female',
 'Qualifies for reduced/free lunch',
 61.0,
 74.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'GJ8SE',
 'Male',
 'Qualifies for reduced/free lunch',
 63.0,
 77.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'GYLNC',
 'Female',
 'Does not qualify',
 62.0,
 82.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'H1H52',
 'Male',
 'Does not qualify',
 69.0,
 81.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'JPSF7',
 'Male',
 'Qualifies for reduced/free lunch',
 63.0,
 75.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
```

```
 'KW69V',
 'Male',
 'Qualifies for reduced/free lunch',
 62.0,
 81.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'MY8XT',
 'Female',
 'Qualifies for reduced/free lunch',
 63.0,
 77.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'NKEK5',
 'Male',
 'Qualifies for reduced/free lunch',
 59.0,
 74.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'PCELB',
 'Male',
 'Does not qualify',
 63.0,
 80.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'VLKJU',
 'Male',
 'Does not qualify',
 66.0,
 76.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'WI916',
 'Female',
 'Does not qualify',
 59.0,
 77.0],
['CCAAW',
```

```
 'Suburban',
 'Non-public',
 'IQN',
 'Experimental',
 15.0,
 'WO5E2',
 'Male',
 'Qualifies for reduced/free lunch',
 60.0,
 74.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'PGK',
 'Standard',
 21.0,
 '0U8DI',
 'Female',
 'Qualifies for reduced/free lunch',
 68.0,
 76.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'PGK',
 'Standard',
 21.0,
 '18YOG',
 'Male',
 'Qualifies for reduced/free lunch',
 74.0,
 82.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'PGK',
 'Standard',
 21.0,
 '4YN68',
 'Female',
 'Qualifies for reduced/free lunch',
 63.0,
 73.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'PGK',
 'Standard',
 21.0,
 '905ZT',
 'Female',
 'Qualifies for reduced/free lunch',
 73.0,
 78.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'PGK',
 'Standard',
 21.0,
 '9WYVP',
```

```
 'Male',
 'Does not qualify',
 75.0,
 83.0],
['CCAAW',
 'Suburban',
 'Non-public',
 'PGK',
 'Standard',
 21.0,
 'A1554',
 'Female',
 'Does not qualify',
 78.0,
 84.0]]
```

In [14]:
```
# Add some sample data
mycol = mydb["customers"]

mylist = [
    {'Country':'United States','First Name' :'Marshall','Last Name': 'Berna
    {'Country':'Ghana', 'First Name' :'Celinda','Last Name':  'Malkin', 'Te
    {'Country':'Ukraine', 'First Name' :'Guillermo','Last Name': 'Furze','
    {'Country':'Greece','First Name' :'Aharon','Last Name': 'Tunnow', 'Test
    {'Country':'Russia', 'First Name' :'Bail', 'Last Name': 'Goodwin', 'Tes
    {'Country':'Poland', 'First Name' :'Cole','Last Name':  'Winteringham',
    {'Country':'Sweden', 'First Name' :'Emlyn', 'Last Name': 'Erricker','Te
    {'Country':'Russia','First Name' : 'Cathee', 'Last Name': 'Sivewright',
    {'Country':'China', 'First Name' :'Barny', 'Last Name': 'Ingerson','Tes
    {'Country':'Uganda','First Name' : 'Sharla', 'Last Name': 'Papaccio',
]
```

In [15]:
```
x = mycol.insert_many(mylist)
```

In [16]:
```
#print list of the _id values of the inserted documents:
print(x.inserted_ids)
```

```
[ObjectId('60ddbb84dd473efd9b0c27b3'), ObjectId('60ddbb84dd473efd9b0c27b4')
, ObjectId('60ddbb84dd473efd9b0c27b5'), ObjectId('60ddbb84dd473efd9b0c27b6'
), ObjectId('60ddbb84dd473efd9b0c27b7'), ObjectId('60ddbb84dd473efd9b0c27b8
'), ObjectId('60ddbb84dd473efd9b0c27b9'), ObjectId('60ddbb84dd473efd9b0c27b
a'), ObjectId('60ddbb84dd473efd9b0c27bb'), ObjectId('60ddbb84dd473efd9b0c27
bc')]
```

Nivel 2

Exercises 1

Load some simple queries to a Pandas Dataframe.

In [17]:
```
# select data from a collection
x1 = mycol.find_one()
print(x1)
```

```
{'_id': ObjectId('60ddbb84dd473efd9b0c27b3'), 'Country': 'United States', '
First Name': 'Marshall', 'Last Name': 'Bernadot', 'Test Score': 54}
```

In [19]:
```
x2 = mycol.find()
entries = list(x2)
entries[:5]
```

Out[19]:
```
[{'_id': ObjectId('60ddbb84dd473efd9b0c27b3'),
  'Country': 'United States',
  'First Name': 'Marshall',
  'Last Name': 'Bernadot',
  'Test Score': 54},
 {'_id': ObjectId('60ddbb84dd473efd9b0c27b4'),
  'Country': 'Ghana',
  'First Name': 'Celinda',
  'Last Name': 'Malkin',
  'Test Score': 51},
 {'_id': ObjectId('60ddbb84dd473efd9b0c27b5'),
  'Country': 'Ukraine',
  'First Name': 'Guillermo',
  'Last Name': 'Furze',
  'Test Score': 53},
 {'_id': ObjectId('60ddbb84dd473efd9b0c27b6'),
  'Country': 'Greece',
  'First Name': 'Aharon',
  'Last Name': 'Tunnow',
  'Test Score': 48},
 {'_id': ObjectId('60ddbb84dd473efd9b0c27b7'),
  'Country': 'Russia',
  'First Name': 'Bail',
  'Last Name': 'Goodwin',
  'Test Score': 46}]
```

In [43]:
```
df_scores = pd.DataFrame(entries)
df_scores
```

Out[43]:

	_id	Country	First Name	Last Name	Test Score
0	60ddbb84dd473efd9b0c27b3	United States	Marshall	Bernadot	54
1	60ddbb84dd473efd9b0c27b4	Ghana	Celinda	Malkin	51
2	60ddbb84dd473efd9b0c27b5	Ukraine	Guillermo	Furze	53
3	60ddbb84dd473efd9b0c27b6	Greece	Aharon	Tunnow	48
4	60ddbb84dd473efd9b0c27b7	Russia	Bail	Goodwin	46
5	60ddbb84dd473efd9b0c27b8	Poland	Cole	Winteringham	49
6	60ddbb84dd473efd9b0c27b9	Sweden	Emlyn	Erricker	55
7	60ddbb84dd473efd9b0c27ba	Russia	Cathee	Sivewright	49
8	60ddbb84dd473efd9b0c27bb	China	Barny	Ingerson	57
9	60ddbb84dd473efd9b0c27bc	Uganda	Sharla	Papaccio	55

In [23]:
```
#Find document(s) with the country "Greece"

myquery_1 = { "Country": "Greece" }

mydoc = mycol.find(myquery_1)

df_list1 = []
for x in mydoc:
    df_list1.append(x)


df_scores1 = pd.DataFrame(df_list1)
df_scores1
```

Out[23]:

	_id	Country	First Name	Last Name	Test Score
0	60ddbb84dd473efd9b0c27b6	Greece	Aharon	Tunnow	48

In [24]:
```
#Find documents where the First Name starts with the letter "G" or higher

myquery_2 = { "First Name": { "$gt": "G" } }

mydoc = mycol.find(myquery_2)

df_list2 = []
for x in mydoc:
    df_list2.append(x)

df_scores2 = pd.DataFrame(df_list2)
df_scores2
```

Out[24]:

	_id	Country	First Name	Last Name	Test Score
0	60ddbb84dd473efd9b0c27b3	United States	Marshall	Bernadot	54
1	60ddbb84dd473efd9b0c27b5	Ukraine	Guillermo	Furze	53
2	60ddbb84dd473efd9b0c27bc	Uganda	Sharla	Papaccio	55

Nivel 3

Exercise 1

Generates a statistical summary of the information contained in the database

In this task, I will use other dataset because there are sufficient entries on it for the exercise.

In [25]:
```
df_scores.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   _id         10 non-null     object
 1   Country     10 non-null     object
 2   First Name  10 non-null     object
 3   Last Name   10 non-null     object
 4   Test Score  10 non-null     int64 
dtypes: int64(1), object(4)
memory usage: 528.0+ bytes
```

In [46]:
```
# How many unique countries there are in the dataframe
df_scores['Country'].nunique()
```

Out[46]:
```
9
```

In [47]:
```
# How many entries per country
df_scores['Country'].value_counts()
```

Out[47]:
```
Russia           2
Poland           1
Sweden           1
United States    1
Uganda           1
Greece           1
Ukraine          1
Ghana            1
China            1
Name: Country, dtype: int64
```

In [48]:
```
score_mean = pd.DataFrame(df_scores.groupby('Country')['Test Score'].mean(
score_mean
```

Out[48]:

	Country	Test Score
0	China	57.0
1	Ghana	51.0
2	Greece	48.0
3	Poland	49.0
4	Russia	47.5
5	Sweden	55.0
6	Uganda	55.0
7	Ukraine	53.0
8	United States	54.0

In [72]:
```
print('The maximum average score is', score_mean['Test Score'].max())
print('The country with the highest average score is',
      score_mean.at[score_mean['Test Score'].idxmax(),'Country'])
```

```
The maximum average score is 57.0
The country with the highest average score is China
```

We can see that China has the highest average test score of this dataframe.

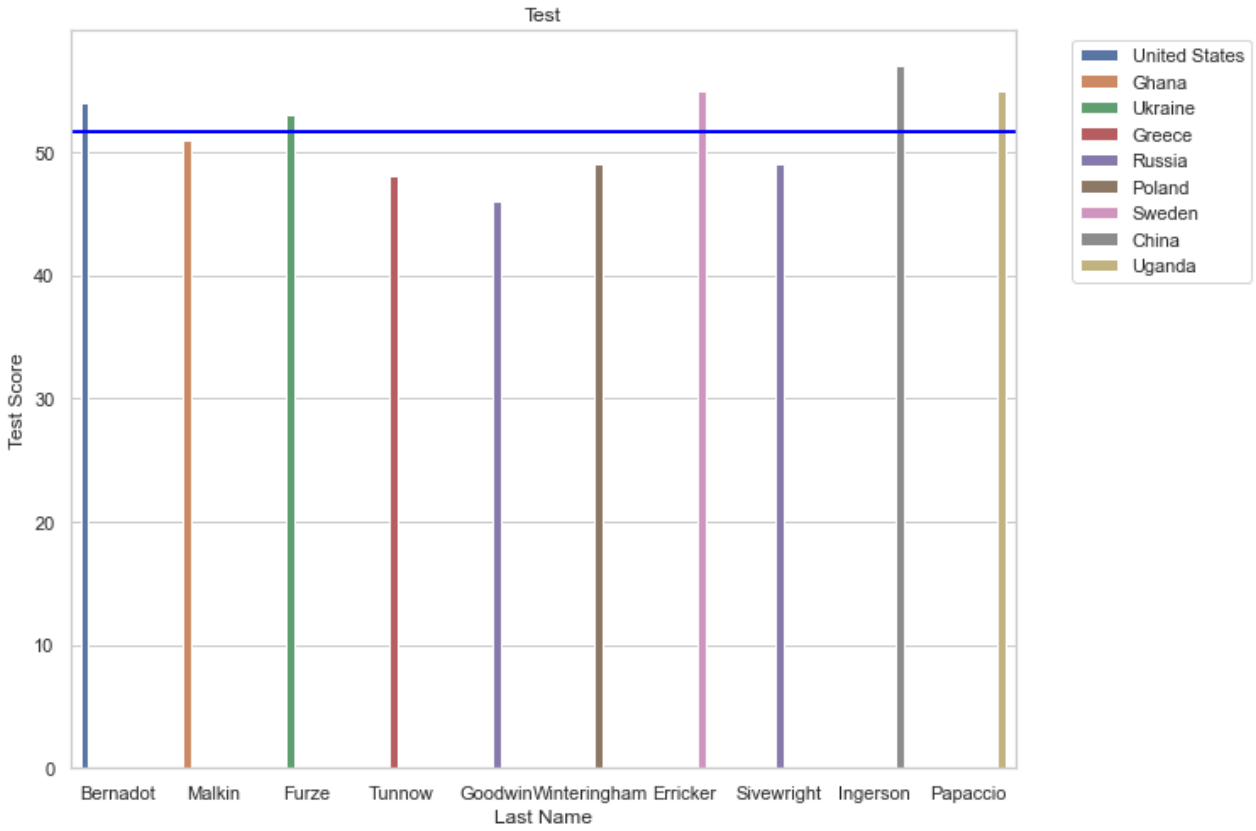
In [70]:
```
score_mean.at[score_mean['Test Score'].idxmax(),'Country']
```

Out[70]:
```
'China'
```

In [73]:
```
# Average test score of all students
df_scores['Test Score'].mean()
```

Out[73]:
```
51.7
```

In [66]:
```
plt.figure(figsize = (10,8))
sns.set_theme(style="whitegrid")
sns.barplot(x='Last Name', y ='Test Score',hue="Country", data = df_scores
plt.legend(bbox_to_anchor=(1.05, 1.0),loc='upper left')
plt.title('Test')
plt.axhline(df_scores['Test Score'].mean(), color='blue', linewidth=2);
```



Individually, the largest note belongs to Ingerson of China, while the smallest belongs to Goodwin of Russia.

We can also see that 50% of students are above the average of 51.7.

In [ ]:
```

```