PANDAS 2; read and SQLite

February 15, 2022

1 Reading files

1.1 Examples

[1]:
```
import requests

download_url = "https://raw.githubusercontent.com/fivethirtyeight/data/master/
 ↪nba-elo/nbaallelo.csv"
target_csv_path = "nba_all_elo.csv"

response = requests.get(download_url)
response.raise_for_status()     # Check that the request was successful
with open(target_csv_path, "wb") as f:
    f.write(response.content)
print("Download ready.")
```

```
Download ready.
```

[3]:
```
import pandas as pd
nba = pd.read_csv("nba_all_elo.csv")
type(nba)
```

[3]:
```
pandas.core.frame.DataFrame
```

[4]:
```
nba.shape
```

[4]:
```
(126314, 23)
```

[5]:
```
nba.head()
```

[5]:

	gameorder	game_id	lg_id	_iscopy	year_id	date_game	seasongame	\
0	1	194611010TRH	NBA	0	1947	11/1/1946	1	
1	1	194611010TRH	NBA	1	1947	11/1/1946	1	
2	2	194611020CHS	NBA	0	1947	11/2/1946	1	

```
3          2  194611020CHS   NBA        1     1947  11/2/1946           2
4          3  194611020DTF   NBA        0     1947  11/2/1946           1

   is_playoffs team_id  fran_id  ...  win_equiv  opp_id  opp_fran  opp_pts  \
0            0     TRH  Huskies  ...  40.294830     NYK    Knicks       68
1            0     NYK   Knicks  ...  41.705170     TRH   Huskies       66
2            0     CHS    Stags  ...  42.012257     NYK    Knicks       47
3            0     NYK   Knicks  ...  40.692783     CHS     Stags       63
4            0     DTF  Falcons  ...  38.864048     WSC  Capitols       50

   opp_elo_i  opp_elo_n  game_location  game_result  forecast notes
0  1300.0000  1306.7233              H            L  0.640065   NaN
1  1300.0000  1293.2767              A            W  0.359935   NaN
2  1306.7233  1297.0712              H            W  0.631101   NaN
3  1300.0000  1309.6521              A            L  0.368899   NaN
4  1300.0000  1320.3811              H            L  0.640065   NaN

[5 rows x 23 columns]
```

[6]:
```
pd.set_option("display.max.columns", None)
```

[7]:
```
pd.set_option("display.precision", 2)
```

[8]:
```
nba.tail()
```

[8]:
```
        gameorder       game_id lg_id  _iscopy  year_id  date_game  \
126309      63155  201506110CLE   NBA        0     2015  6/11/2015
126310      63156  201506140GSW   NBA        0     2015  6/14/2015
126311      63156  201506140GSW   NBA        1     2015  6/14/2015
126312      63157  201506170CLE   NBA        0     2015  6/16/2015
126313      63157  201506170CLE   NBA        1     2015  6/16/2015

        seasongame  is_playoffs team_id    fran_id  pts    elo_i    elo_n  \
126309         100            1     CLE  Cavaliers   82  1723.41  1704.39
126310         102            1     GSW   Warriors  104  1809.98  1813.63
126311         101            1     CLE  Cavaliers   91  1704.39  1700.74
126312         102            1     CLE  Cavaliers   97  1700.74  1692.09
126313         103            1     GSW   Warriors  105  1813.63  1822.29

        win_equiv opp_id   opp_fran  opp_pts  opp_elo_i  opp_elo_n  \
126309      60.31    GSW   Warriors      103    1790.96    1809.98
126310      68.01    CLE  Cavaliers       91    1704.39    1700.74
126311      60.01    GSW   Warriors      104    1809.98    1813.63
126312      59.29    GSW   Warriors      105    1813.63    1822.29
126313      68.52    CLE  Cavaliers       97    1700.74    1692.09

       game_location game_result  forecast notes
```

```
126309             H           L      0.55   NaN
126310             H           W      0.77   NaN
126311             A           L      0.23   NaN
126312             H           L      0.48   NaN
126313             A           W      0.52   NaN
```

[47]:
```
imdb_data = pd.read_csv("IMDB-Movie-Data.csv")
imdb_data.head(10)
```

[47]:
```
   Rank                    Title                        Genre  \
0     1  Guardians of the Galaxy     Action,Adventure,Sci-Fi
1     2               Prometheus    Adventure,Mystery,Sci-Fi
2     3                    Split             Horror,Thriller
3     4                     Sing     Animation,Comedy,Family
4     5           Suicide Squad    Action,Adventure,Fantasy
5     6          The Great Wall    Action,Adventure,Fantasy
6     7               La La Land          Comedy,Drama,Music
7     8                 Mindhorn                       Comedy
8     9      The Lost City of Z  Action,Adventure,Biography
9    10               Passengers     Adventure,Drama,Romance

                                         Description             Director  \
0  A group of intergalactic criminals are forced ...           James Gunn
1  Following clues to the origin of mankind, a te...          Ridley Scott
2  Three girls are kidnapped by a man with a diag...    M. Night Shyamalan
3  In a city of humanoid animals, a hustling thea...  Christophe Lourdelet
4  A secret government agency recruits some of th...           David Ayer
5  European mercenaries searching for black powde...          Yimou Zhang
6  A jazz pianist falls for an aspiring actress i...       Damien Chazelle
7  A has-been actor best known for playing the ti...            Sean Foley
8  A true-life drama, centering on British explor...           James Gray
9  A spacecraft traveling to a distant colony pla...         Morten Tyldum

                                        Actors  Year  Runtime (Minutes)  \
0  Chris Pratt, Vin Diesel, Bradley Cooper, Zoe S...  2014                121
1  Noomi Rapace, Logan Marshall-Green, Michael Fa...  2012                124
2  James McAvoy, Anya Taylor-Joy, Haley Lu Richar...  2016                117
3  Matthew McConaughey,Reese Witherspoon, Seth Ma...  2016                108
4  Will Smith, Jared Leto, Margot Robbie, Viola D...  2016                123
5      Matt Damon, Tian Jing, Willem Dafoe, Andy Lau  2016                103
6  Ryan Gosling, Emma Stone, Rosemarie DeWitt, J...  2016                128
7  Essie Davis, Andrea Riseborough, Julian Barrat...  2016                 89
8  Charlie Hunnam, Robert Pattinson, Sienna Mille...  2016                141
9  Jennifer Lawrence, Chris Pratt, Michael Sheen,...  2016                116

   Rating   Votes  Revenue (Millions)  Metascore
0     8.1  757074              333.13       76.0
```

```
1     7.0  485820              126.46       65.0
2     7.3  157606              138.12       62.0
3     7.2   60545              270.32       59.0
4     6.2  393727              325.02       40.0
5     6.1   56036               45.13       42.0
6     8.3  258682              151.06       93.0
7     6.4    2490                 NaN       71.0
8     7.1    7188                8.01       78.0
9     7.0  192177              100.01       41.0
```

[49]:
```
Billboard = pd.read_csv("charts.csv")
Billboard.head()
```

[49]:
```
         date  rank           song                         artist  last-week  \
0  2021-11-06     1     Easy On Me                          Adele        1.0
1  2021-11-06     2           Stay  The Kid LAROI & Justin Bieber        2.0
2  2021-11-06     3  Industry Baby       Lil Nas X & Jack Harlow        3.0
3  2021-11-06     4     Fancy Like                  Walker Hayes        4.0
4  2021-11-06     5     Bad Habits                    Ed Sheeran        5.0

   peak-rank  weeks-on-board
0          1               3
1          1              16
2          1              14
3          3              19
4          2              18
```

1.2 SQLite and pandas

[5]:
```
import sqlite3
from sqlite3 import Error


def create_connection(db_file):
    """ create a database connection to a SQLite database """
    conn = None
    try:
        conn = sqlite3.connect(db_file)
        print(sqlite3.version)
    except Error as e:
        print(e)
    finally:
        if conn:
            conn.close()
```

```
if __name__ == '__main__':
    create_connection(r"C:\sqlite\db\pythonsqlite.db")
```

```
2.6.0
```

You have to connect to the database by passing the database file to `connect()` function. If the database does not exist then new database is created and you are connected to that database. And if in case the database exists then you are connected to that database.

[30]:
```
import sqlite3

conn = sqlite3.connect('mobiledevices.db')
print('Connected to database successfully.')
```

```
Connected to database successfully.
```

[31]:
```
conn = sqlite3.connect(':memory')
print('Connected to database successfully.')
```

```
Connected to database successfully.
```

Get Cursor Object from Connection Next, use a connection.cursor() method to create a cursor object. using cursor object we can execute SQL queries.

[32]:
```
cur = conn.cursor()
```

Creation of a Table if not exists already:

[36]:
```
cur.execute("""CREATE TABLE IF NOT EXISTS AndroidPhones (
    id  INTEGER NOT NULL,
    brand   TEXT,
    model   TEXT,
    os  TEXT,
    cpu TEXT,
    PRIMARY KEY(`id`)
)""")
print('Table created successfully.')

conn.commit()
```

```
Table created successfully.
```

Next, prepare a SQL INSERT query to insert a row into a table. in the insert query, we mention column names and their values to insert in a table. For example, INSERT INTO mysql_table (column1, column2, . . .) VALUES (value1, value2, . . .);

[34]:
```
phone1 = ("Samsung", "Galaxy A7 2018", "Android v8.0 Oreo", "Octa core 2.2 GHz")
phone2 = ["LG", "G7 Fit", "Android v8.1 Oreo", "Quad core 2.15 GHz"]
```

```
phone3 = {"brand":"Motorola", "model":"G6 Plus", "os":"Android v8.0 Oreo", "cpu":
 ↪"Octa core 2.2 GHz"}

cur.execute("insert into androidphones(brand, model, os, cpu) values(?, ?, ?, ?
 ↪)", phone1)
cur.execute("insert into androidphones(brand, model, os, cpu) values(?, ?, ?, ?
 ↪)", phone2)
cur.execute("insert into androidphones(brand, model, os, cpu) values(:brand, :
 ↪model, :os, :cpu)", phone3)
print('Records inserted successfully.')

conn.commit()
```

```
Records inserted successfully.
```

[37]:
```
phone1 = (
   ("Samsung", "Galaxy S10 Lite", "Android v9.0 Pie", "Octa core"),
   ("HTC", "Desire 12s", "Android v8.1 Oreo", "Quad core 1.4 GHz")
)
phone2 = [
   ["HTC", "Exodus 1", "Android v8.1 Oreo", "Octa core 2.8 GHz"],
   ["Motorola", "G7", "Android v9.0 Pie", "Octa core 2.2 GHz"]
]
phone3 = [
   ("Motorola", "P30 Note", "Android v8.0 Oreo", "Octa core 1.8 GHz"),
   ("Xiaomi", "Redmi 7 Pro", "Android v8.1 Oreo", "Octa core 2 GHz")
]

cur.executemany("insert into androidphones(brand, model, os, cpu) values(?, ?, ?
 ↪, ?)", phone1)
cur.executemany("insert into androidphones(brand, model, os, cpu) values(?, ?, ?
 ↪, ?)", phone2)
cur.executemany("insert into androidphones(brand, model, os, cpu) values(?, ?, ?
 ↪, ?)", phone3)
print('Records inserted successfully.')

conn.commit()
```

```
Records inserted successfully.
```

[38]:
```
phone1 = ("Galaxy J4 Core", "Android v8.1 Oreo", "Quad core 1.4 GHz", 4)

cur.execute("update androidphones set model=?, os=?, cpu=? where id=?", phone1)
print('Records updated successfully.')

conn.commit()
```

```
Records updated successfully.
```

[41]:
```
rows = cur.execute("select * from androidphones")
for row in rows:
    print("ID: "+str(row[0]))
    print("Brand: "+row[1])
    print("Model: "+row[2])
    print("O.S.: "+row[3])
    print("C.P.U.: "+row[4])
    print("")
```

```
ID: 1
Brand: Samsung
Model: Galaxy S10 Lite
O.S.: Android v9.0 Pie
C.P.U.: Octa core

ID: 2
Brand: HTC
Model: Desire 12s
O.S.: Android v8.1 Oreo
C.P.U.: Quad core 1.4 GHz

ID: 3
Brand: HTC
Model: Exodus 1
O.S.: Android v8.1 Oreo
C.P.U.: Octa core 2.8 GHz

ID: 4
Brand: Motorola
Model: Galaxy J4 Core
O.S.: Android v8.1 Oreo
C.P.U.: Quad core 1.4 GHz

ID: 5
Brand: Motorola
Model: P30 Note
O.S.: Android v8.0 Oreo
C.P.U.: Octa core 1.8 GHz

ID: 6
Brand: Xiaomi
Model: Redmi 7 Pro
O.S.: Android v8.1 Oreo
C.P.U.: Octa core 2 GHz
```

[45]:
```
import pandas as pd

df = pd.read_sql_query("select * from androidphones", conn)
```

[46]:
```
print(df)
```

```
   id     brand            model                  os               cpu
0   1   Samsung  Galaxy S10 Lite   Android v9.0 Pie         Octa core
1   2       HTC      Desire 12s  Android v8.1 Oreo  Quad core 1.4 GHz
2   3       HTC        Exodus 1  Android v8.1 Oreo  Octa core 2.8 GHz
3   4  Motorola  Galaxy J4 Core  Android v8.1 Oreo  Quad core 1.4 GHz
4   5  Motorola        P30 Note  Android v8.0 Oreo  Octa core 1.8 GHz
5   6    Xiaomi     Redmi 7 Pro  Android v8.1 Oreo    Octa core 2 GHz
```

[ ]: