TAVOLE STATISTICHE

```
import sqlite3, pandas as pd, requests, os, sys, sqlalchemy, duckdb
from io import BytesIO
import json
from pyjstat import pyjstat
```

```
conn = sqlite3.connect("D:/files/Bankit.sqlite")
from sqlalchemy import create_engine
sqlite = create_engine('sqlite:///D:/files/Bankit.sqlite')

def carica_dati_in_sql(tabella):
    dtypes = {
        "DESINV": sqlalchemy.types.INTEGER(),
        "DURORI": sqlalchemy.types.INTEGER(),
        "TIPTASSO": sqlalchemy.types.INTEGER(),
        "VALORE": sqlalchemy.types.INTEGER(),
        "CLASSE_ACCORD":sqlalchemy.types.TEXT()
    }
    data.to_sql(tabella, sqlite, if_exists='replace', index=False, dtype=dtypes)
    return

file_path = 'D:\\DatiStatistici.xlsx'
# file_path = 'C:\\Users\\PVolterr\\Mediocredito Centrale S.p.A\\Studi e Governo Iniziative -
from openpyxl import load_workbook
# os.chdir('C:\\Users\\PVolterr\\Mediocredito Centrale S.p.A\\Studi e Governo Iniziative - D
from sqlalchemy import create_engine

db_file = 'D:/files/Bankit.duckdb' # Nome del file del database
ddb = duckdb.connect(db_file) # Connessione al database DuckDB (crea il file se non esiste)

query = "SELECT * FROM `domain-stacoris-multicube`"
stacoris = pd.read_sql_query(query, conn)
query = "SELECT * FROM `domain-stafinra-multicube`"
```

```
stafinra = pd.read_sql_query(query, conn)
query = "SELECT * FROM `domain-stamen-multicube`"
stamen = pd.read_sql_query(query, conn)

nuts1 = ['IT','ITC','ITC1','ITC2','ITC3','ITC4','ITH','ITH3','ITH4','ITH5','ITHBI12','ITI','I
```

```
ddb.execute(f"DROP TABLE IF EXISTS {tabella}")
ddb.execute(f"CREATE TABLE IF NOT EXISTS {tabella} AS SELECT * FROM data LIMIT 0")  # Crea u
ddb.execute(f"INSERT INTO {tabella} SELECT * FROM data")
```

```
<duckdb.duckdb.DuckDBPyConnection at 0x1a6fae1d2f0>
```

```
sheet_to_update = tabella
book = load_workbook(file_path)
with pd.ExcelWriter(file_path, engine='openpyxl', mode='a', if_sheet_exists='replace') as wri
    data.to_excel(writer, sheet_name=sheet_to_update, index=False)
```

TDB10266 DEPOSITS | Distribution by customer location (geographical area) and branch of economic activity

```
tabella = 'TDB10266'
file = f'https://a2a.bancaditalia.it/infostat/dataservices/export/IT/CSV/DATA/CUBE/BANKITALI
result = requests.get(file)
date_column = ['DATA_OSS']
data = pd.read_csv(BytesIO(result.content),compression='zip', header=0, sep=';', quotechar='
data['DATA_OSS'] = pd.to_datetime(data['DATA_OSS'])
data
```

	DATA_OSS	DIVISA1	DURORI	ENTE_SEGN	FENEC	LOC_CTP	RAMATECO	RESIDI
0	2008-09-30	1000	9	1100010	1041810	IT	51	IT
1	2008-09-30	1000	9	1100010	1041810	IT	52	IT
2	2008-09-30	1000	9	1100010	1041810	IT	53	IT
3	2008-09-30	1000	9	1100010	1041810	IT	54	IT
4	2008-09-30	1000	9	1100010	1041810	IT	55	IT
...	...	...	...	...	...	...	...	...
6187	1998-03-31	1000	9	1100010	1041810	ITI	70	IT
6188	1998-03-31	1000	9	1100010	1041810	ITI	71	IT
6189	1998-03-31	1000	9	1100010	1041810	ITI	72	IT

	DATA_OSS	DIVISA1	DURORI	ENTE_SEGN	FENEC	LOC_CTP	RAMATECO	RESIDE
6190	1998-03-31	1000	9	1100010	1041810	ITI	73	IT
6191	1998-03-31	1000	9	1100010	1041810	ITI	4999	IT

```
data['DATA_OSS'].max()
```

```
Timestamp('2008-09-30 00:00:00')
```

```
# data = data[data['DATA_OSS'] == data['DATA_OSS'].max()]
# data['DATA_OSS'] = data['DATA_OSS'].dt.date
# data = data[data['LOC_CTP'].isin(nuts1)]
data = pd.merge(data, stamen, how = 'left', left_on='ENTE_SEGN', right_on='Elemento').drop(co
data = pd.merge(data, stamen, how = 'left', left_on='LOC_CTP', right_on='Elemento').drop(colu
data = pd.merge(data, stamen, how = 'left', left_on='SET_CTP', right_on='Elemento').drop(colu
# data = data[['DATA_OSS', 'LOC_CTP', 'area', 'SET_CTP','target','VALORE']]
dtypes = {"DIVISA1": sqlalchemy.types.INTEGER(), "DURORI": sqlalchemy.types.INTEGER(),"LOC_SE
         "VALORE": sqlalchemy.types.INTEGER()}
data.to_sql('TDB10266', sqlite, if_exists='replace', dtype=dtypes,index=False)
```

```
sheet_to_update = tabella
book = load_workbook(file_path)
with pd.ExcelWriter(file_path, engine='openpyxl', mode='a', if_sheet_exists='replace') as wri
    data.to_excel(writer, sheet_name=sheet_to_update, index=False)
```

TDB10290

```
tabella = 'TDB10290'
file = f'https://a2a.bancaditalia.it/infostat/dataservices/export/IT/CSV/DATA/CUBE/BANKITALIA
result = requests.get(file)
date_column = ['DATA_OSS']
data = pd.read_csv(BytesIO(result.content),compression='zip', header=0, sep=';', quotechar='"
data['DATA_OSS'] = pd.to_datetime(data['DATA_OSS'])
data = data[data['DATA_OSS'] == data['DATA_OSS'].max()]
data['DATA_OSS'] = data['DATA_OSS'].dt.date
data = data[data['LOC_CTP'].isin(nuts1)]
data = pd.merge(data, stamen, how = 'left', left_on='ENTE_SEGN', right_on='Elemento').drop(co
data = pd.merge(data, stamen, how = 'left', left_on='LOC_CTP', right_on='Elemento').drop(colu
```

```
data = pd.merge(data, stamen, how = 'left', left_on='SET_CTP', right_on='Elemento').drop(colu
data = data[['DATA_OSS', 'LOC_CTP', 'area', 'SET_CTP','target','VALORE']]
sheet_to_update = tabella
book = load_workbook(file_path)
with pd.ExcelWriter(file_path, engine='openpyxl', mode='a', if_sheet_exists='replace') as wri
    data.to_excel(writer, sheet_name=sheet_to_update, index=False)
```

```
tabella = 'TDB10295'
file = f'https://a2a.bancaditalia.it/infostat/dataservices/export/IT/CSV/DATA/CUBE/BANKITALIA
result = requests.get(file)
date_column = ['DATA_OSS']
data = pd.read_csv(BytesIO(result.content),compression='zip', header=0, sep=';', quotechar='"
data['DATA_OSS'] = pd.to_datetime(data['DATA_OSS'])
data = data[data['DATA_OSS'] == data['DATA_OSS'].max()]
data['DATA_OSS'] = data['DATA_OSS'].dt.date
data = data[data['LOC_CTP'].isin(nuts1)]
data = pd.merge(data, stamen, how = 'left', left_on='ENTE_SEGN', right_on='Elemento').drop(co
data = pd.merge(data, stamen, how = 'left', left_on='LOC_CTP', right_on='Elemento').drop(colu
data = pd.merge(data, stamen, how = 'left', left_on='SET_CTP', right_on='Elemento').drop(colu
data = data[['DATA_OSS', 'LOC_CTP', 'area', 'SET_CTP','target','VALORE']]
sheet_to_update = tabella
book = load_workbook(file_path)
with pd.ExcelWriter(file_path, engine='openpyxl', mode='a', if_sheet_exists='replace') as wri
    data.to_excel(writer, sheet_name=sheet_to_update, index=False)
```

```
tabella = 'TDB20290'
file = f'https://a2a.bancaditalia.it/infostat/dataservices/export/IT/CSV/DATA/CUBE/BANKITALIA
result = requests.get(file)
date_column = ['DATA_OSS']
data = pd.read_csv(BytesIO(result.content),compression='zip', header=0, sep=';', quotechar='"
data['DATA_OSS'] = pd.to_datetime(data['DATA_OSS'])
#data = data[data['DATA_OSS'] == data['DATA_OSS'].max()]
data['DATA_OSS'] = data['DATA_OSS'].dt.date
#data = data[data['LOC_CTP'].isin(nuts1)]
data = pd.merge(data, stamen, how = 'left', left_on='ENTE_SEGN', right_on='Elemento').drop(co
data = pd.merge(data, stamen, how = 'left', left_on='LOC_CTP', right_on='Elemento').drop(colu
data = pd.merge(data, stamen, how = 'left', left_on='SET_CTP', right_on='Elemento').drop(colu
data = data[['DATA_OSS', 'LOC_CTP', 'area', 'SET_CTP','target','VALORE']]
sheet_to_update = tabella
book = load_workbook(file_path)
with pd.ExcelWriter(file_path, engine='openpyxl', mode='a', if_sheet_exists='replace') as wri
    data.to_excel(writer, sheet_name=sheet_to_update, index=False)
```

tabella = ‘TFR10255’ file = f’https://a2a.bancaditalia.it/infostat/dataservices/export/IT/CSV/DATA/CUBE/ result = requests.get(file) date_column = [‘DATA_OSS’] data = pd.read_csv(BytesIO(result.content),compress header=0, sep=‘;’, quotechar=“‘, encoding=‘utf-8’,dtype={‘ENTE_SEGN’:‘str’, ‘FENEC’:‘str’, ‘VALORE’:‘Int32’,‘LOC_SPORT’:‘Int32’},parse_dates=date_column, dayfirst=False) data[‘DATA_OSS’] = pd.to_datetime(data[‘DATA_OSS’]) data = data[data[‘DATA_OSS’] == data[‘DATA_OSS’].max()] data[‘DATA_OSS’] = data[‘DATA_OSS’].dt.date data = data[data[‘LOC_CTP’].isin(nuts1)] data = pd.merge(data, stamen, how = ‘left’, left_on=‘ENTE_SEGN’, right_on=‘Elemento’).drop(columns=[‘ENTE_SEGN’,‘STATUS’,‘FENEC’,‘index’,‘I ‘segnalante’}) data = pd.merge(data, stamen, how = ‘left’, left_on=‘LOC_CTP’, right_on=‘Elemento’).drop(columns=[‘index’,‘Dominio’, ‘Elemento’]).rename(columns={‘Descrizione’: ‘area’}) data = pd.merge(data, stamen, how = ‘left’, left_on=‘SET_CTP’, right_on=‘Elemento’).drop(columns ‘Elemento’]).rename(columns={‘Descrizione’: ‘target’}) data = pd.merge(data, stamen, how = ‘left’, left_on=‘ATECO_CTP’, right_on=‘Elemento’).drop(columns=[‘index’,‘Dominio’, ‘Elemento’]).rename(columns={‘Descrizione’: ‘ATECO’}) data = data[[‘DATA_OSS’, ‘LOC_CTP’, ‘area’, ‘ATECO_CTP’, ‘ATECO’,‘SET_CTP’,‘target’,‘VALORE’]] sheet_to_update = tabella book = load_workbook(file_path) with pd.ExcelWriter(file_path, engine=‘openpyxl’, mode=‘a’, if_sheet_exists=‘replace’) as writer: data.to_excel(writer, sheet_name=sheet_to_update, index=False)

```
tabella = 'TFR20231'
file = f'https://a2a.bancaditalia.it/infostat/dataservices/export/IT/CSV/DATA/CUBE/BANKITALI
result = requests.get(file)
date_column = ['DATA_OSS']
data = pd.read_csv(BytesIO(result.content),compression='zip', header=0, sep=';', quotechar='
data['DATA_OSS'] = pd.to_datetime(data['DATA_OSS'])
data = data[data['DATA_OSS'] == data['DATA_OSS'].max()]
data['DATA_OSS'] = data['DATA_OSS'].dt.date
data = data[data['LOC_CTP'].isin(nuts1)]
data = pd.merge(data, stamen, how = 'left', left_on='ENTE_SEGN', right_on='Elemento').drop(co
data = pd.merge(data, stamen, how = 'left', left_on='LOC_CTP', right_on='Elemento').drop(colu
data = pd.merge(data, stamen, how = 'left', left_on='SET_CTP', right_on='Elemento').drop(colu
#data = pd.merge(data, stamen, how = 'left', left_on='ATECO_CTP', right_on='Elemento').drop(
data = data[['DATA_OSS', 'LOC_CTP', 'area', 'SET_CTP','target','VALORE']] # 'ATECO_CTP', 'ATE
sheet_to_update = tabella
book = load_workbook(file_path)
with pd.ExcelWriter(file_path, engine='openpyxl', mode='a', if_sheet_exists='replace') as wri
    data.to_excel(writer, sheet_name=sheet_to_update, index=False)
```

```
tabella = 'TRI30603'
file = f'https://a2a.bancaditalia.it/infostat/dataservices/export/IT/CSV/DATA/CUBE/BANKITALI
result = requests.get(file)
date_column = ['DATA_OSS']
data = pd.read_csv(BytesIO(result.content),compression='zip', header=0, sep=';', quotechar='
```

```
data['DATA_OSS'] = pd.to_datetime(data['DATA_OSS'])
data = data[data['DATA_OSS'] == data['DATA_OSS'].max()]
data['DATA_OSS'] = data['DATA_OSS'].dt.date
data = data[data['SEDELEG_SOGG'].isin(nuts1)]
data = pd.merge(data, stamen, how = 'left', left_on='ENTE_SEGN', right_on='Elemento').drop(co
data = pd.merge(data, stamen, how = 'left', left_on='SEDELEG_SOGG', right_on='Elemento').drop
data = pd.merge(data, stamen, how = 'left', left_on='SET_CTP', right_on='Elemento').drop(colu
#data = pd.merge(data, stamen, how = 'left', left_on='ATECO_CTP', right_on='Elemento').drop(c
data = data[['DATA_OSS', 'SEDELEG_SOGG', 'area', 'SET_CTP','target','VALORE']] # 'ATECO_CTP'
sheet_to_update = tabella
book = load_workbook(file_path)
with pd.ExcelWriter(file_path, engine='openpyxl', mode='a', if_sheet_exists='replace') as wri
    data.to_excel(writer, sheet_name=sheet_to_update, index=False)
```