Ζήκος Σπύρος 1084581, 3ο έτος

Αρχές Γλωσσών Προγραμματισμού και Μεταφραστών

Προτζεκτ Python, Ιούνιος 2023

Κώδικας

```
import requests, csv
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
import numpy as np
import mysql.connector
from tkinter import *


def arr2csv(arr, fname):   # given a 2d array and a filename: make a csv
    with open(fname, 'w', newline='') as f:
        writer = csv.writer(f)
        for i in arr:
            writer.writerow(i)


def download_csv():    # download the data to a local file in the same directory as
the python file
    r =
requests.get('https://www.stats.govt.nz/assets/Uploads/Effects-of-COVID-19-on-tra
de/Effects-of-COVID-19-on-trade-At-15-December-2021-provisional/Download-data/eff
ects-of-covid-19-on-trade-at-15-december-2021-provisional.csv')

    with open("data.csv", "wb") as f:
        f.writelines(r)


def create_db():   # create a mysql database
    with open("mysqlconfig.txt","r") as f:   # get credentials
        host1,user1,pwd1 = f.readline().split(",")
        host,user,pwd =
[host1.split("=")[1][1:-1],user1.split("=")[1][1:-1],pwd1.split("=")[1][1:-1]]
```

```
    mydb = mysql.connector.connect(host=host, user=user, passwd=pwd)
    mycursor = mydb.cursor()

    try:
        mycursor.execute("CREATE DATABASE effects_of_covid_on_trade")   # if the db
doesn't exist
    except:
        mycursor.execute("DROP DATABASE effects_of_covid_on_trade")   # if it
already exists delete and remake
        mycursor.execute("CREATE DATABASE effects_of_covid_on_trade")

    mydb = mysql.connector.connect(host=host, user=user,
passwd=pwd,database="effects_of_covid_on_trade")
    mycursor = mydb.cursor()

    return (mydb, mycursor)


def create_tables():
    mycursor.execute("CREATE TABLE r_per_month (month VARCHAR(20) PRIMARY KEY,
dollars BIGINT, tonnes BIGINT)")
    mycursor.execute("CREATE TABLE r_per_country (country VARCHAR(100) PRIMARY KEY,
dollars BIGINT, tonnes BIGINT)")
    mycursor.execute("CREATE TABLE r_per_transport_mode (transport_mode
VARCHAR(100) PRIMARY KEY, dollars BIGINT, tonnes BIGINT)")
    mycursor.execute("CREATE TABLE r_per_day (day VARCHAR(20) PRIMARY KEY, dollars
BIGINT, tonnes BIGINT)")
    mycursor.execute("CREATE TABLE r_per_commodity (commodity VARCHAR(100) PRIMARY
KEY, dollars BIGINT, tonnes BIGINT)")

    mycursor.execute("CREATE TABLE mr_per_5months (month VARCHAR(20), revenue
BIGINT, measure VARCHAR(40), PRIMARY KEY (month, measure))")
    mycursor.execute("CREATE TABLE mr_per_5commodities (country VARCHAR(100),
revenue BIGINT, measure VARCHAR(40), PRIMARY KEY (country, measure))")
    mycursor.execute("CREATE TABLE mr_day_per_commodity (commodity VARCHAR(100),
day VARCHAR(20), value BIGINT, measure VARCHAR(40), PRIMARY KEY (commodity,
measure))")


def ins(val, tname, key):   # used for the first 5 plots: put data in database table
and csv
    sql = f"INSERT INTO {tname} ({key}, dollars, tonnes) VALUES (%s, %s, %s)"
    mycursor.executemany(sql, val)
    mydb.commit()
    # make the headers for the csv
    if key=="month":
        headers = [("Month","Dollars","Tonnes")]
    elif key=="country":
        headers = [("Country","Dollars","Tonnes")]
    elif key=="transport_mode":
        headers = [("Transport mode","Dollars","Tonnes")]
    elif key=="day":
        headers = [("Weekday","Dollars","Tonnes")]
    elif key=="commodity":
        headers = [("Commodity","Dollars","Tonnes")]
    # make the csv
    arr2csv(headers+val, str(tname)+".csv")

def ins2(val, tname, key):   # used for the 6 and 7 plots
    sql = f"INSERT INTO {tname} ({key}, revenue, measure) VALUES (%s, %s, %s)"
    mycursor.executemany(sql, val)
    mydb.commit()
    if key=="month":
        headers = [("Month","Revenue","Measure")]
    elif key=="country":
        headers = [("Country","Revenue","Measure")]
    arr2csv(headers+val, str(tname)+".csv")

def ins3(val):    # used for the last plot
    sql = f"INSERT INTO mr_day_per_commodity (commodity, day, value, measure) VALUES
(%s, %s, %s, %s)"
    mycursor.executemany(sql, val)
    mydb.commit()
    headers = [("Commodity","Weekday","Value", "Measure")]
    arr2csv(headers+val, "mr_day_per_commodity.csv")


def total_revenue_per_month(dby, chart):
    df['Date'] = pd.to_datetime(df['Date'])
    df['Month'] = df['Date'].dt.strftime('%m')   # make a 'Month' column

    months =
['Jan','Feb',"Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"]

    if dby:   # if dby==True: insert the data in the right table and a csv
        dollars = df[df["Measure"]=="$"].groupby(['Month'])["Value"].sum()
        tonnes = df[df["Measure"]=="Tonnes"].groupby(['Month'])["Value"].sum()
        ins(list(zip(months,dollars,tonnes)), "r_per_month", "month")
    if chart:   # if chart==True: plot the data after some approximations
        dollars =
df[df["Measure"]=="$"].groupby(['Month'])["Value"].sum()/10000//1e5
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Month'])["Value"].sum()//1e5

        x = np.arange(len(months))  # list -> 0 - 11
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(layout='constrained')

        r = ax.bar(x, dollars, width, label='Dollars In Billions')
        ax.bar_label(r, padding=3)
        r = ax.bar(x + width, tonnes, width, label='Tonnes In Hundred of Thousands')
        ax.bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax.set_xlabel('Months')
        ax.set_ylabel('Total revenue')
        ax.set_title('Total revenue per month')
        ax.set_xticks(x + width, months)
        ax.legend(loc='lower right')

        plt.show()


def total_revenue_per_country(dby, chart):
    dollars =
df[df["Measure"]=="$"].groupby(['Country'])["Value"].sum()/10000//1e5
    countries = []   # get all the countries
    for i in range(len(dollars)):
        countries.append(dollars.index[i])

    if dby:
        dollars = df[df["Measure"]=="$"].groupby(['Country'])["Value"].sum()
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Country'])["Value"].sum()   # does not
contain all countries
        tonnes2 = df[df["Measure"]=="$"].groupby(['Country'])["Value"].sum()  #
placeholder
        for i in countries:
            try:
                tonnes2[i] = tonnes[i]
            except:
                tonnes2[i] = 0
        ins(list(zip(countries,dollars,tonnes2)), "r_per_country", "country")
    if chart:
        dollars =
df[df["Measure"]=="$"].groupby(['Country'])["Value"].sum()/10000//1e5
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Country'])["Value"].sum()//1e5
        tonnes2 = df[df["Measure"]=="$"].groupby(['Country'])["Value"].sum()  #
placeholder
        for i in countries:
            try:
                tonnes2[i] = tonnes[i]
            except:
                tonnes2[i] = 0

        x = np.arange(len(countries))  # list -> 0 - 8
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(layout='constrained')

        r = ax.bar(x, dollars, width, label='Dollars In Billions')
        ax.bar_label(r, padding=3)
        r = ax.bar(x + width, tonnes2, width, label='Tonnes In Hundred of Thousands')
        ax.bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax.set_xlabel('Country')
        ax.set_ylabel('Total revenue')
        ax.set_title('Total revenue per country')
        ax.set_xticks(x + width, countries)
        ax.tick_params('x', labelsize='small')
        ax.legend(loc='upper right')

        plt.show()


def total_revenue_per_transport_mode(dby, chart):
    dollars =
df[df["Measure"]=="$"].groupby(['Transport_Mode'],sort=False)["Value"].sum()/1000
0//1e5
    modes = []
    for i in range(len(dollars)):
        modes.append(dollars.index[i])

    if dby:
        dollars =
df[df["Measure"]=="$"].groupby(['Transport_Mode'],sort=False)["Value"].sum()
```

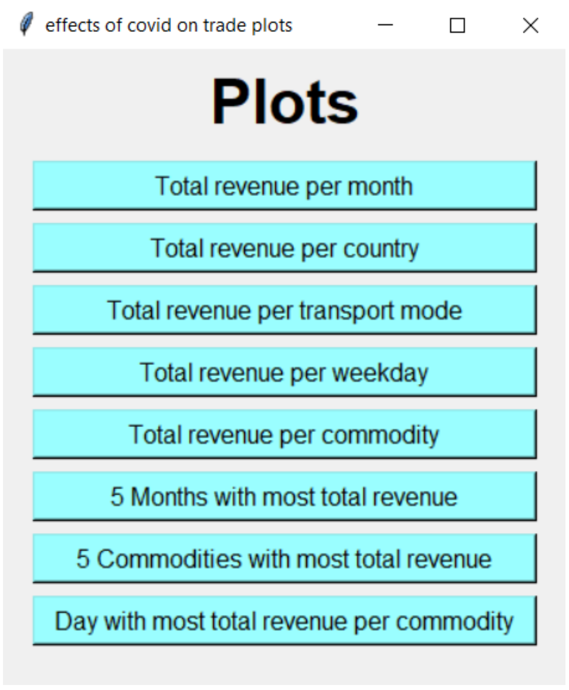
```
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Transport_Mode'],sort=False)["Value"].sum()
        tonnes2 =
df[df["Measure"]=="$"].groupby(['Transport_Mode'],sort=False)["Value"].sum()  #
placeholder
        for i in modes:
            try:
                tonnes2[i] = tonnes[i]
            except:
                tonnes2[i] = 0
        ins(list(zip(modes,dollars,tonnes2)), "r_per_transport_mode",
"transport_mode")

    if chart:
        dollars =
df[df["Measure"]=="$"].groupby(['Transport_Mode'],sort=False)["Value"].sum()/1000
0//1e5
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Transport_Mode'],sort=False)["Value"].sum()
//1e5
        tonnes2 =
df[df["Measure"]=="$"].groupby(['Transport_Mode'],sort=False)["Value"].sum()  #
placeholder
        for i in modes:
            try:
                tonnes2[i] = tonnes[i]
            except:
                tonnes2[i] = 0

        x = np.arange(len(modes))  # list -> 0 - 3
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(layout='constrained')

        r = ax.bar(x, dollars, width, label='Dollars In Billions')
        ax.bar_label(r, padding=3)
        r = ax.bar(x + width, tonnes2, width, label='Tonnes In Hundred of Thousands')
        ax.bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax.set_xlabel('Transport mode')
        ax.set_ylabel('Total revenue')
        ax.set_title('Total revenue per transport mode')
        ax.set_xticks(x + width, modes)
        ax.tick_params('x', labelsize='small')
        ax.legend(loc='upper right')
        plt.show()


def total_revenue_per_day(dby, chart):
    days = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday",
"Sunday"]

    if dby:
        dollars =
df[df["Measure"]=="$"].groupby(['Weekday'],sort=False)["Value"].sum()
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Weekday'],sort=False)["Value"].sum()
        dollars2, tonnes2 = [[], []]
        for i in days:
            dollars2.append(int(dollars[i]))
            tonnes2.append(int(tonnes[i]))
        ins(list(zip(days,dollars2,tonnes2)), "r_per_day", "day")

    if chart:
        dollars =
df[df["Measure"]=="$"].groupby(['Weekday'],sort=False)["Value"].sum()/10000//1e5
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Weekday'],sort=False)["Value"].sum()//1e5

        dollars2, tonnes2 = [[], []]
        for i in days:
            dollars2.append(dollars[i])
            tonnes2.append(tonnes[i])

        x = np.arange(len(days))  # list -> 0 - 6
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(layout='constrained')

        r = ax.bar(x, dollars2, width, label='Dollars In Billions')
        ax.bar_label(r, padding=3)
        r = ax.bar(x + width, tonnes2, width, label='Tonnes In Hundred of Thousands')
        ax.bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax.set_xlabel('Weekday')
        ax.set_ylabel('Total revenue')
        ax.set_title('Total revenue per weekday')
        ax.set_xticks(x + width, days)
        ax.tick_params('x', labelsize='small')
        ax.legend(loc='lower right')

        plt.show()


def total_revenue_per_commodity(dby, chart):
    dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()/10000//1e
5
    commodities = []
    for i in range(len(dollars)):
        commodities.append(dollars.index[i])

    if dby:
        dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Commodity'],sort=False)["Value"].sum()
        tonnes2 =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()  #
placeholder
        for i in commodities:
            try:
                tonnes2[i] = int(tonnes[i])
            except:
                tonnes2[i] = 0
        ins(list(zip(commodities,dollars,tonnes2)), "r_per_commodity",
"commodity")

    if chart:
        dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()/10000//1e
5
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Commodity'],sort=False)["Value"].sum()//1e5
        tonnes2 =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()/10000//1e
5  # placeholder

        for i in commodities:
            try:
                tonnes2[i] = tonnes[i]
            except:
                tonnes2[i] = 0
        x = np.arange(len(commodities))  # list -> 0 - 8
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(layout='constrained')

        r = ax.bar(x, dollars, width, label='Dollars In Billions')
        ax.bar_label(r, padding=3)
        r = ax.bar(x + width, tonnes2, width, label='Tonnes In Hundred of Thousands')
        ax.bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax.set_xlabel('Commodity')
        ax.set_ylabel('Total revenue')
        ax.set_title('Total revenue per commodity')
        ax.set_xticks(x + width, commodities)
        ax.tick_params('x', labelsize='xx-small')
        ax.legend(loc='upper right')

        plt.show()


def most_revenue_5_months(dby, chart):
    df['Date'] = pd.to_datetime(df['Date'])
    df['Month'] = df['Date'].dt.strftime('%m')

    months =
['Jan','Feb',"Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"]
    month_dic = {}   # { 1:'Jan', 2:'Feb', ... }
    for i in range(12):
        month_dic[i+1] = months[i]

    if dby:
        dollars =
df[df["Measure"]=="$"].groupby(['Month'],sort=True)["Value"].sum().nlargest(5)
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Month'],sort=True)["Value"].sum().nlargest(
5)

        months_dol,months_ton = [[], []]
        for i in range(len(dollars)):
            months_dol.append(month_dic[int(dollars.index[i])])
        for i in range(len(dollars)):
            months_ton.append(month_dic[int(tonnes.index[i])])

        dol,ton = [[], []]   # for the function zip
        for i in range(len(months_ton)):
            dol.append("$")
            ton.append("Tonnes")
        ins2(list(zip(months_dol,dollars,dol))+list(zip(months_ton,tonnes,ton)),
"mr_per_5months", "month")

    if chart:
        dollars =
df[df["Measure"]=="$"].groupby(['Month'],sort=True)["Value"].sum().nlargest(5)/10
000//1e5
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Month'],sort=True)["Value"].sum().nlargest(
5)//1e5

        months_dol,months_ton = [[], []]
        for i in range(len(dollars)):
            months_dol.append(month_dic[int(dollars.index[i])])
        for i in range(len(dollars)):
            months_ton.append(month_dic[int(tonnes.index[i])])

        x = np.arange(len(range(5)))  # list -> 0 - 4
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(1, 2, layout='constrained')

        r = ax[0].bar(x, dollars, width, label='Dollars In Billions')
        ax[0].bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax[0].set_xlabel('Months')
        ax[0].set_ylabel('Total revenue')
        ax[0].set_title('5 Months with most total revenue')
        ax[0].set_xticks(x, months_dol)
        ax[0].legend(loc='lower right')

        r = ax[1].bar(x, tonnes, width, label='Tonnes In Hundred of Thousands',
color='tab:orange')
        ax[1].bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax[1].set_xlabel('Months')
        ax[1].set_ylabel('Total revenue')
        ax[1].set_title('5 Months with most total revenue')
        ax[1].set_xticks(x, months_ton)
        ax[1].legend(loc='lower right')

        plt.show()
```

```
def most_revenue_5_commodities(dby, chart):
    if dby:
        dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=True)["Value"].sum().nlargest(5
)
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Commodity'],sort=True)["Value"].sum().nlarg
est(4)
        tonnes['(Rest)'] = 0

        commodities_dol, commodities_ton = [[], []]
        for i in range(len(dollars)):
            commodities_dol.append(dollars.index[i])
        for i in range(len(dollars)):
            commodities_ton.append(tonnes.index[i])

        dol,ton = [[], []]
        for i in range(len(commodities_dol)):
            dol.append("$")
            ton.append("Tonnes")

        ins2(list(zip(commodities_dol,dollars,dol))+list(zip(commodities_ton,tonn
es,ton)), "mr_per_5commodities", "country")

    if chart:
        dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=True)["Value"].sum().nlargest(5
)/10000//1e5
        tonnes =
df[df["Measure"]=="Tonnes"].groupby(['Commodity'],sort=True)["Value"].sum().nlarg
est(4)//1e5
        tonnes['(Rest)'] = 0

        commodities_dol, commodities_ton = [[], []]
        for i in range(len(dollars)):
            commodities_dol.append(dollars.index[i])
        for i in range(len(dollars)):
            commodities_ton.append(tonnes.index[i])

        x = np.arange(len(range(5)))  # list -> 0 - 4
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(1, 2, layout='constrained')

        r = ax[0].bar(x, dollars, width, label='Dollars In Billions')
        ax[0].bar_label(r, padding=3)
        # labels, title, custom x-axis tick labels and legend
        ax[0].set_xlabel('Months')
        ax[0].set_ylabel('Total revenue')
        ax[0].set_title('5 Commodities with most total revenue')
        ax[0].set_xticks(x, commodities_dol)
        ax[0].tick_params('x', labelsize='xx-small')
        ax[0].legend(loc='upper right')


        r = ax[1].bar(x, tonnes, width, label='Tonnes In Hundred of Thousands',
color='tab:orange')
        ax[1].bar_label(r, padding=3)

        # labels, title, custom x-axis tick labels and legend
        ax[1].set_xlabel('Months')
        ax[1].set_ylabel('Total revenue')
        ax[1].set_title('5 Commodities with most total revenue')
        ax[1].set_xticks(x, commodities_ton)
        ax[1].tick_params('x', labelsize='xx-small')
        ax[1].legend(loc='upper right')

        plt.show()


def most_revenue_weekday_per_commodity(dby, chart):
    dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()/10000//1e
5
    commodities = []
    for i in range(len(dollars)):
        commodities.append(dollars.index[i])

    if dby:
        dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()

        dol, ton, dol_day, ton_day = [[],[],[],[]]
        for i in commodities:
            a = df[(df["Measure"]=="$") &
(df['Commodity']==i)].groupby(['Weekday'],sort=True)["Value"].sum().nlargest(1)
            dol.append(int(a.values[0]))
            dol_day.append(a.index[0][:3])
            try:
                b = df[(df["Measure"]=="Tonnes") &
(df['Commodity']==i)].groupby(['Weekday'],sort=True)["Value"].sum().nlargest(1)
                ton.append(int(b.values[0]))
                ton_day.append(b.index[0][:3])
            except:
                ton.append(0)
                ton_day.append("(None)")

        d1 = []
        t1 = []
        for i in range(len(commodities)):
            d1.append("$")
            t1.append("Tonnes")
        ins3(list(zip(commodities, dol_day, dol, d1))+list(zip(commodities,
ton_day, ton, t1)))

    if chart:
        dollars =
df[df["Measure"]=="$"].groupby(['Commodity'],sort=False)["Value"].sum()/10000//1e
5

        dol, ton, dol_day, ton_day = [[],[],[],[]]
        for i in commodities:
            a = df[(df["Measure"]=="$") &
(df['Commodity']==i)].groupby(['Weekday'],sort=True)["Value"].sum().nlargest(1)/1
0000//1e5
            dol.append(a.values[0])
            dol_day.append(a.index[0][:3])
            try:
                b = df[(df["Measure"]=="Tonnes") &
(df['Commodity']==i)].groupby(['Weekday'],sort=True)["Value"].sum().nlargest(1)//
1e5
                ton.append(b.values[0])
                ton_day.append(b.index[0][:3])
            except:
                ton.append(0)
                ton_day.append("(None)")

        x = np.arange(len(commodities))  # list -> 0 - 8
        width = 0.25  # width of the bars
        fig, ax = plt.subplots(layout='constrained')

        r = ax.bar(x, dol, width, label='Dollars In Billions')
        ax.bar_label(r, padding=3, labels=dol_day)
        r = ax.bar(x + width, ton, width, label='Tonnes In Hundred of Thousands')
        ax.bar_label(r, padding=3, labels=ton_day)

        # labels, title, custom x-axis tick labels and legend
        ax.set_xlabel('Commodities')
        ax.set_ylabel('Total revenue')
        ax.set_title('Day with most total revenue per commodity')
        ax.set_xticks(x + width, commodities)
        ax.tick_params('x', labelsize='xx-small')
        ax.legend(loc='upper right')

        plt.show()


def store_in_db_and_csv():
    total_revenue_per_month(1,0)
    total_revenue_per_country(1,0)
    total_revenue_per_transport_mode(1,0)
    total_revenue_per_day(1,0)
    total_revenue_per_commodity(1,0)
    most_revenue_5_months(1,0)
    most_revenue_5_commodities(1,0)
    most_revenue_weekday_per_commodity(1,0)


def GUI():
    root = Tk(className="Effects of covid on trade plots")

    def trp_month():
        total_revenue_per_month(0,1)

    def trp_country():
        total_revenue_per_country(0,1)

    def trp_transport_mode():
        total_revenue_per_transport_mode(0,1)

    def trp_day():
        total_revenue_per_day(0,1)

    def trp_commodity():
        total_revenue_per_commodity(0,1)

    def mr_5_months():
        most_revenue_5_months(0,1)
    def mr_5_commodities():
        most_revenue_5_commodities(0,1)

    def mr_day_pc():
        most_revenue_weekday_per_commodity(0,1)


    l = Label(root, text="Plots", font='Helvetica 30 bold')
    l.pack(pady=8)

    trpmonth = Button(root, text="Total revenue per month", command=trp_month,
width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(padx=20, pady=4)

    trpmonth = Button(root, text="Total revenue per country", command=trp_country,
width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(pady=4)

    trpmonth = Button(root, text="Total revenue per transport mode",
command=trp_transport_mode, width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(pady=4)

    trpmonth = Button(root, text="Total revenue per weekday", command=trp_day,
width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(pady=4)

    trpmonth = Button(root, text="Total revenue per commodity",
command=trp_commodity, width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(pady=4)

    trpmonth = Button(root, text="5 Months with most total revenue",
command=mr_5_months, width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(pady=4)

    trpmonth = Button(root, text="5 Commodities with most total revenue",
command=mr_5_commodities, width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(pady=4)

    trpmonth = Button(root, text="Day with most total revenue per commodity",
command=mr_day_pc, width=35, font=20, bg='#9AFEFF')
    trpmonth.pack(pady=4)

    l = Label(root, text="")   # a little more space
    l.pack(pady=1)
    root.mainloop()

try:
    mydb, mycursor = create_db()
    create_tables()
except:
    print("The database is not well configured!")

try:
    df = pd.read_csv('data.csv')
except:
    download_csv()
    df = pd.read_csv('data.csv')

try:
    store_in_db_and_csv()
except:
    print("Something went wrong with the database or the csv")

GUI()
```

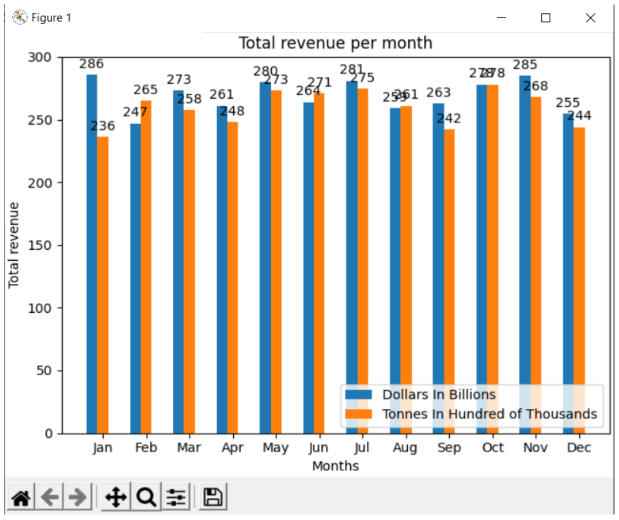
Screenshots παραδειγμάτων της εφαρμογής (και του σχήματος της βάσης δεδομένων)

ΠΑΡΑΔΕΙΓΜΑ 1

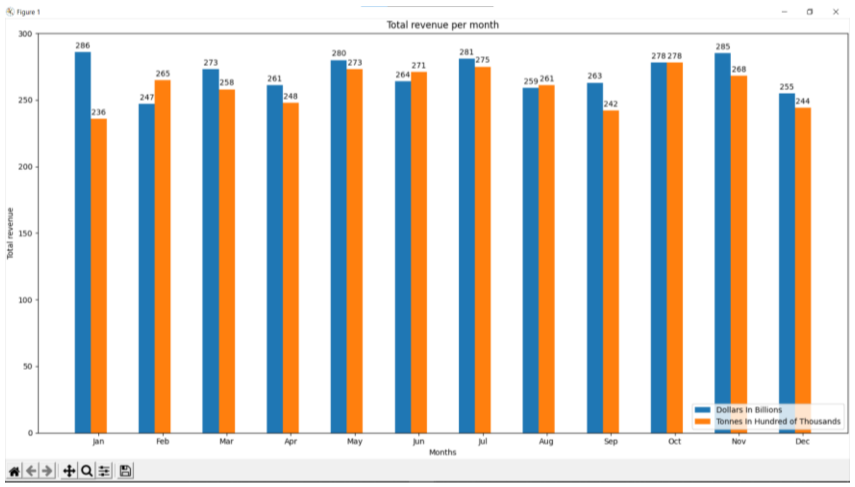
Εκτελούμε το αρχείο python:


effects of covid on trade plots
Plots
Total revenue per month
Total revenue per country
Total revenue per transport mode
Total revenue per weekday
Total revenue per commodity
5 Months with most total revenue
5 Commodities with most total revenue
Day with most total revenue per commodity

Πατάμε στο πρώτο κουμπί:



Μεγενθύνουμε το παράθυρο για να φανεί καλύτερα:

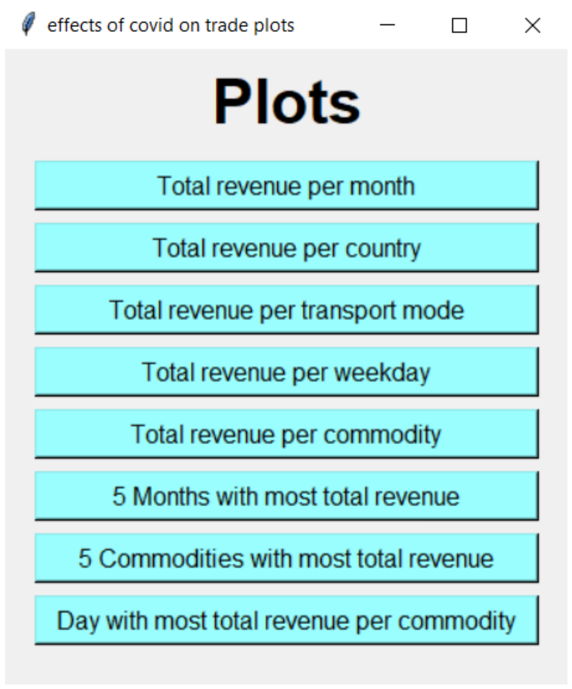


Ύστερα, πατάμε στο x για να κλείσουμε το παράθυρο με το διάγραμμα.

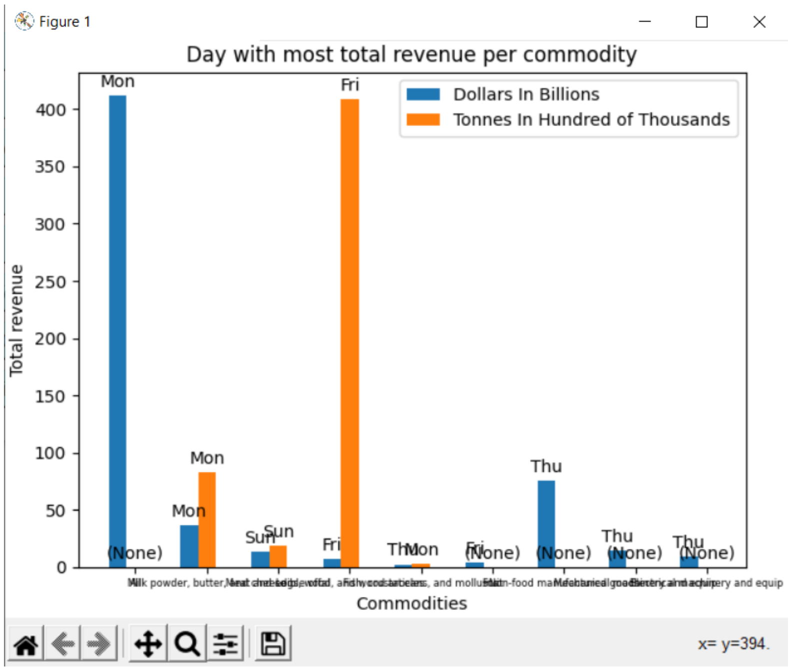
Τέλος, πατάμε στο x για να κλείσουμε το παράθυρο με το μενού.

ΠΑΡΑΔΕΙΓΜΑ 2

Εκτελούμε το αρχείο python:


effects of covid on trade plots
Plots
Total revenue per month
Total revenue per country
Total revenue per transport mode
Total revenue per weekday
Total revenue per commodity
5 Months with most total revenue
5 Commodities with most total revenue
Day with most total revenue per commodity

Πατάμε στο τελευταίο κουμπί:



Μεγενθύνουμε το παράθυρο για να φανεί καλύτερα:

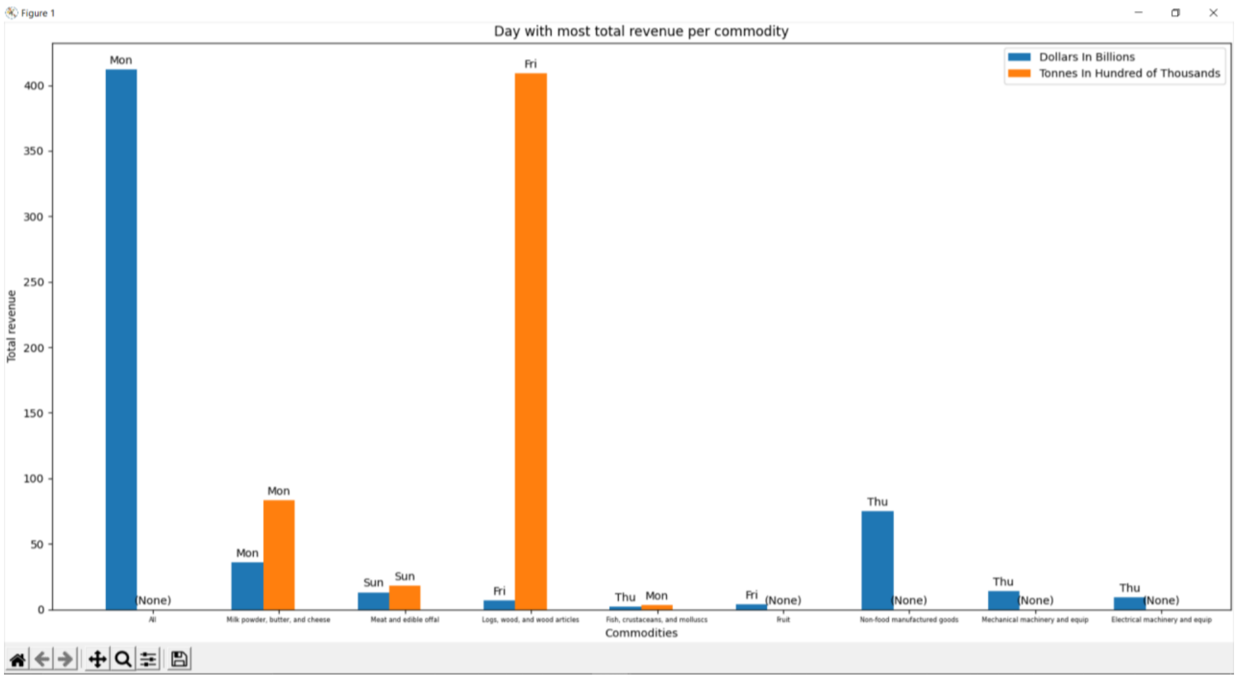
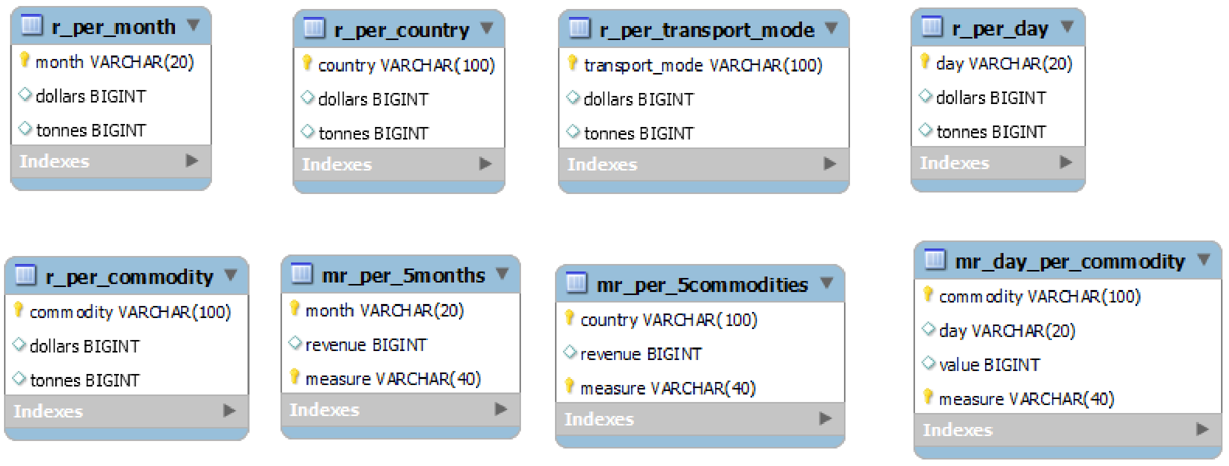


Figure 1
Day with most total revenue per commodity
Dollars In Billions
Tonnes In Hundred of Thousands
Total revenue
0
50
100
150
200
250
300
350
400
Mon
(None)
Mon
Mon
Sun
Sun
Fri
Fri
Thu
Mon
Fri
(None)
Thu
(None)
Thu
(None)
Thu
(None)
All
Milk powder, butter, and cheese
Meat and edible offal
Logs, wood, and wood articles
Fish, crustaceans, and molluscs
Fruit
Non-food manufactured goods
Mechanical machinery and equip
Electrical machinery and equip
Commodities

Ύστερα, πατάμε στο x για να κλείσουμε το παράθυρο με το διάγραμμα.

Τέλος, πατάμε στο x για να κλείσουμε το παράθυρο με το μενού.

ΣΧΗΜΑ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ



r_per_month
- month VARCHAR(20)
- dollars BIGINT
- tonnes BIGINT

r_per_country
- country VARCHAR(100)
- dollars BIGINT
- tonnes BIGINT

r_per_transport_mode
- transport_mode VARCHAR(100)
- dollars BIGINT
- tonnes BIGINT

r_per_day
- day VARCHAR(20)
- dollars BIGINT
- tonnes BIGINT

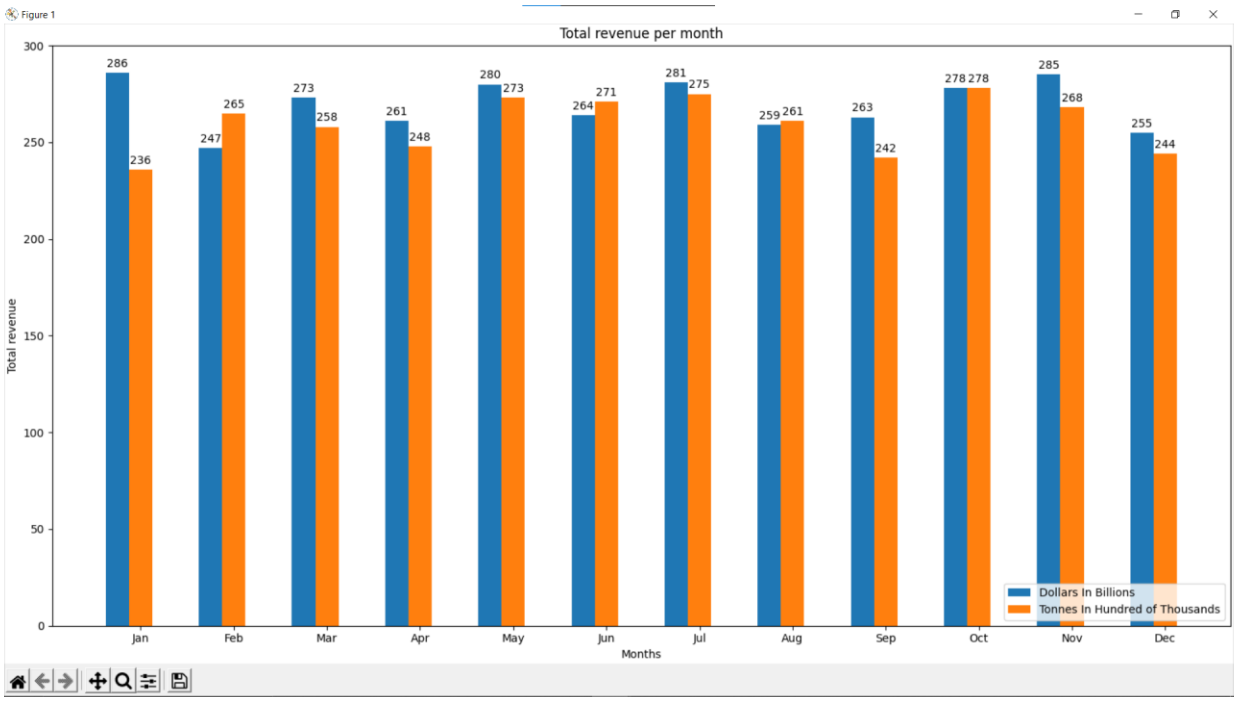
r_per_commodity
- commodity VARCHAR(100)
- dollars BIGINT
- tonnes BIGINT

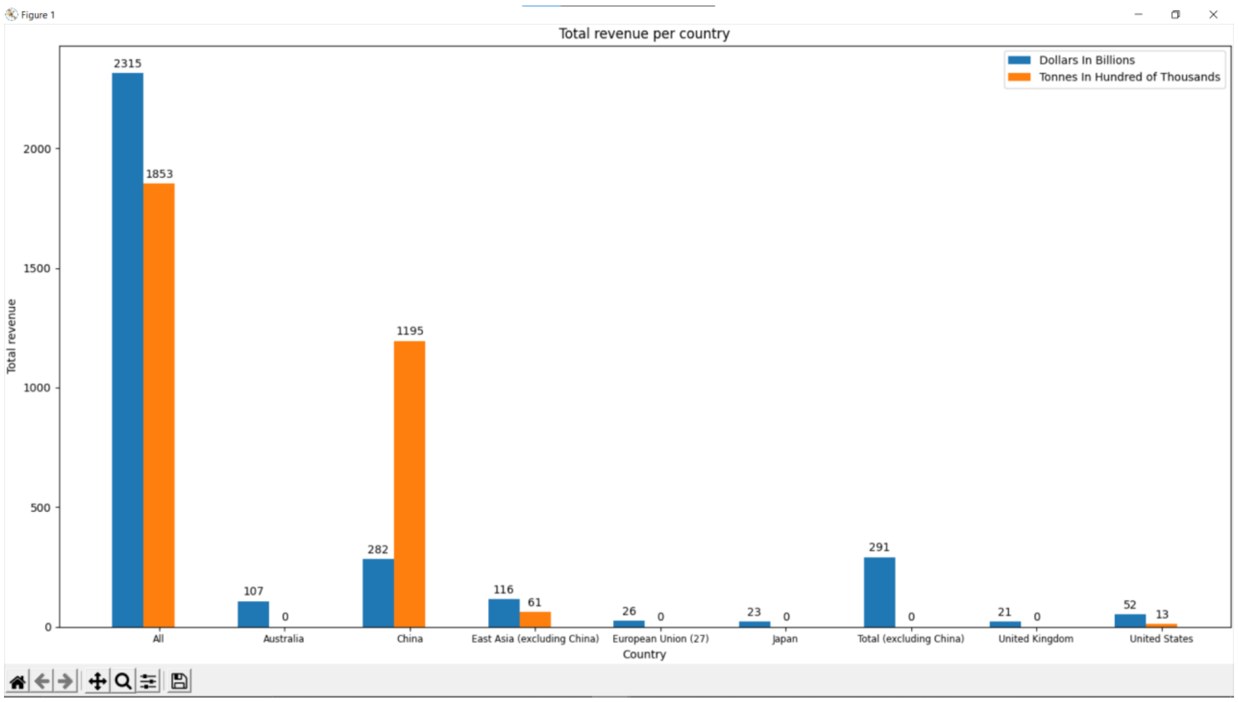
mr_per_5months
- month VARCHAR(20)
- revenue BIGINT
- measure VARCHAR(40)

mr_per_5commodities
- country VARCHAR(100)
- revenue BIGINT
- measure VARCHAR(40)

mr_day_per_commodity
- commodity VARCHAR(100)
- day VARCHAR(20)
- value BIGINT
- measure VARCHAR(40)

Τα ζητούμενα γραφήματα (με τίτλους, υπόμνημα)





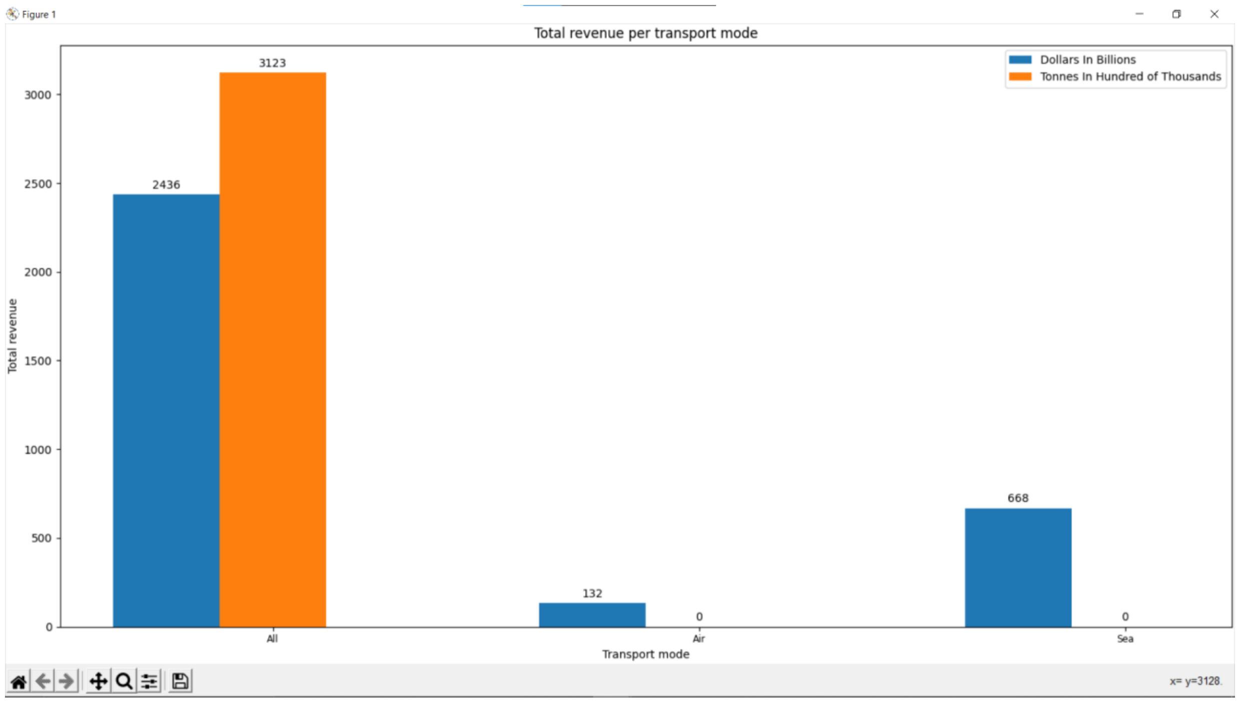


Figure 1
Total revenue per transport mode
Dollars In Billions
Tonnes In Hundred of Thousands
Total revenue
Transport mode
All
Air
Sea
2436
3123
132
0
668
0
0
500
1000
1500
2000
2500
3000
x= y=3128.

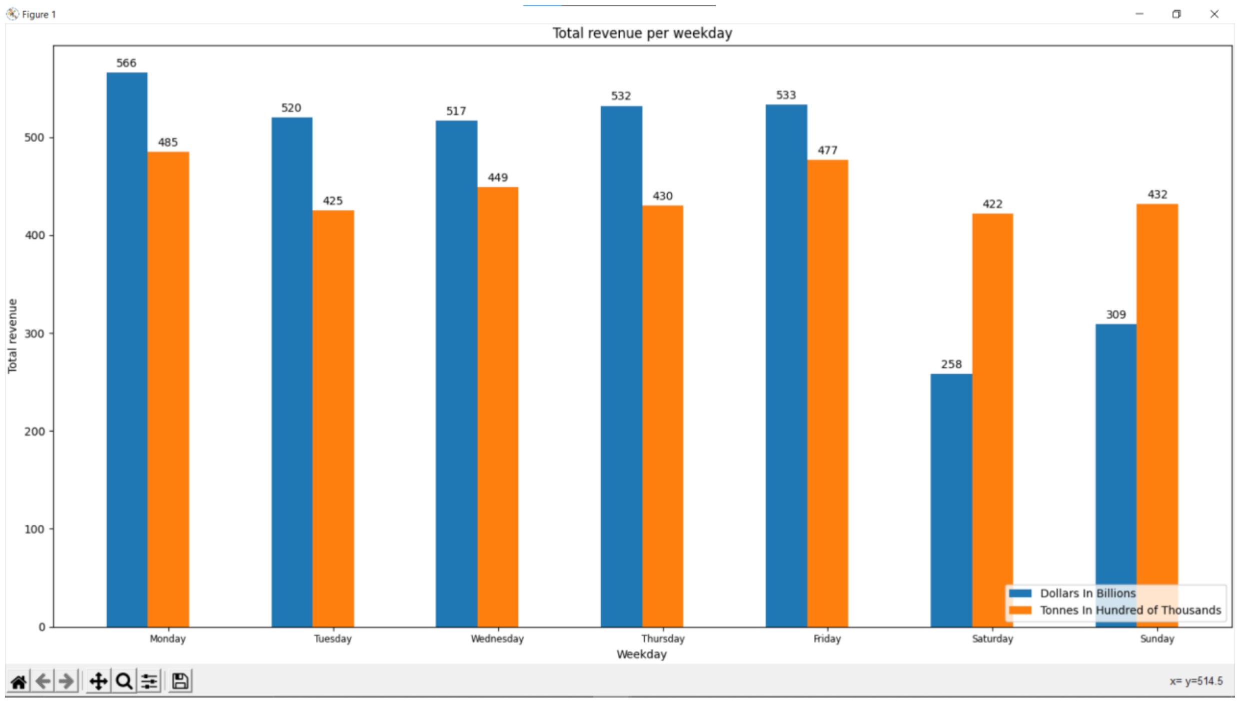


Figure 1
Total revenue per weekday
Dollars In Billions
Tonnes In Hundred of Thousands
Total revenue
Weekday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
566
485
520
425
517
449
532
430
533
477
258
422
309
432
0
100
200
300
400
500
x= y=514.5

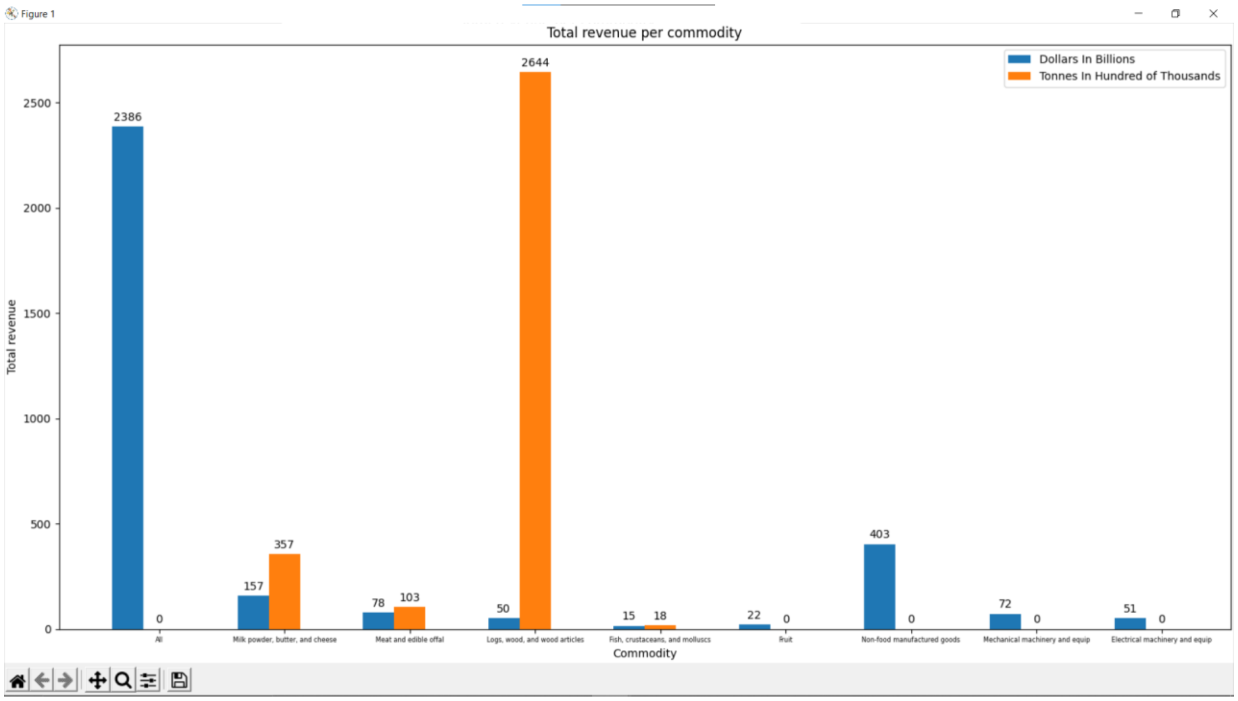


Figure 1
Total revenue per commodity
Dollars In Billions
Tonnes In Hundred of Thousands
Total revenue
Commodity
2386
0
157
357
78
103
50
2644
15
18
22
0
403
0
72
0
51
0
All
Milk powder, butter, and cheese
Meat and edible offal
Logs, wood, and wood articles
Fish, crustaceans, and molluscs
Fruit
Non-food manufactured goods
Mechanical machinery and equip
Electrical machinery and equip

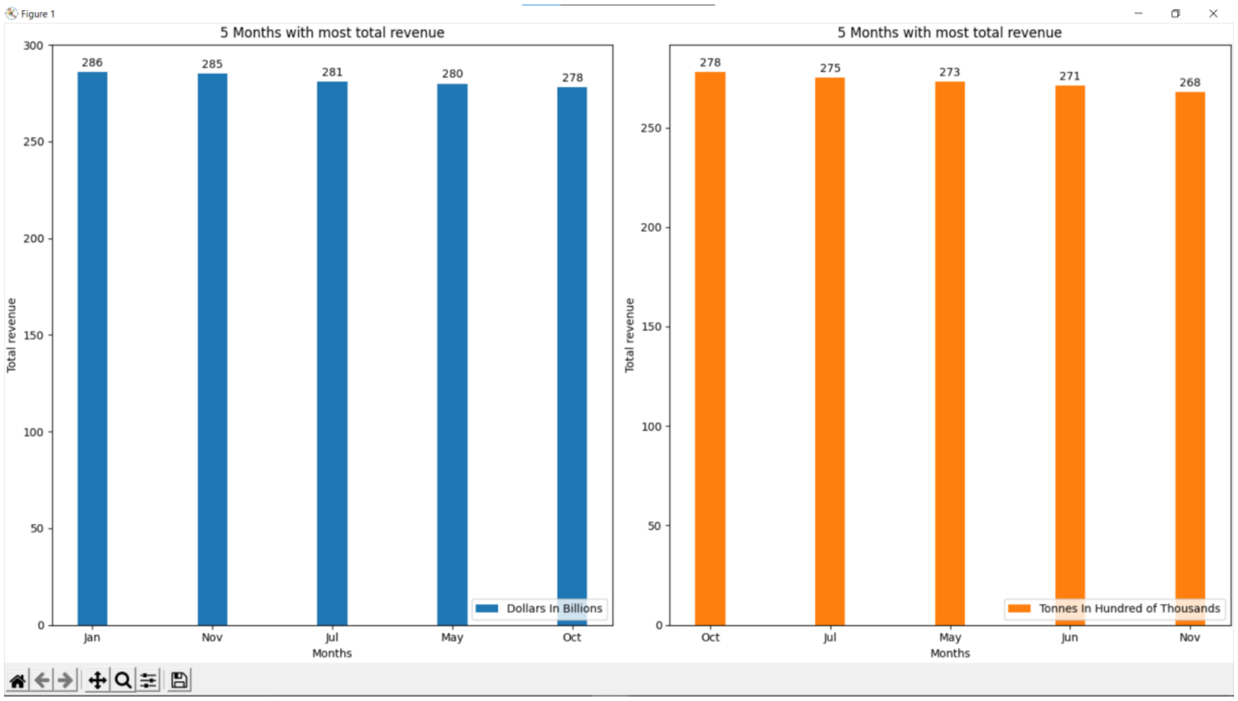


Figure 1
5 Months with most total revenue
Total revenue
Months
Dollars In Billions
286
285
281
280
278
Jan
Nov
Jul
May
Oct
5 Months with most total revenue
Total revenue
Months
Tonnes In Hundred of Thousands
278
275
273
271
268
Oct
Jul
May
Jun
Nov

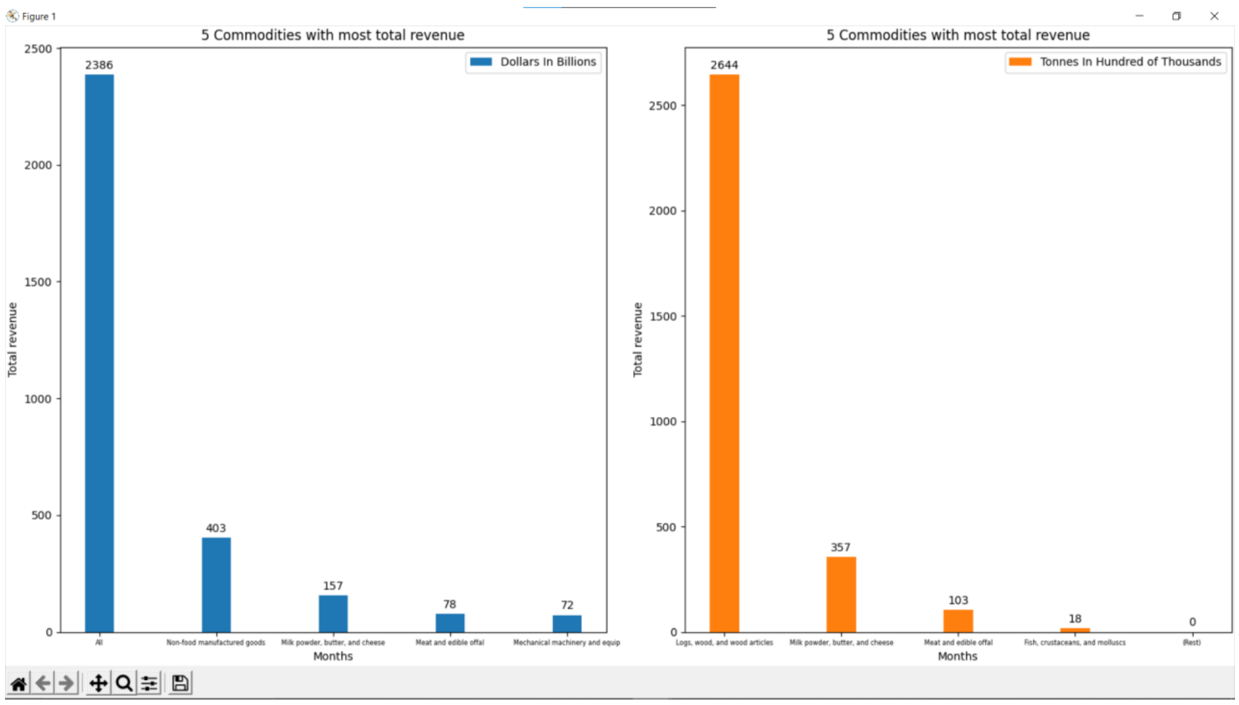


Figure 1
5 Commodities with most total revenue
Dollars In Billions
Total revenue
2500
2000
1500
1000
500
0
2386
403
157
78
72
All
Non-food manufactured goods
Milk powder, butter, and cheese
Meat and edible offal
Mechanical machinery and equip
Months
5 Commodities with most total revenue
Tonnes In Hundred of Thousands
Total revenue
2500
2000
1500
1000
500
0
2644
357
103
18
0
Logs, wood, and wood articles
Milk powder, butter, and cheese
Meat and edible offal
Fish, crustaceans, and molluscs
(Rest)
Months

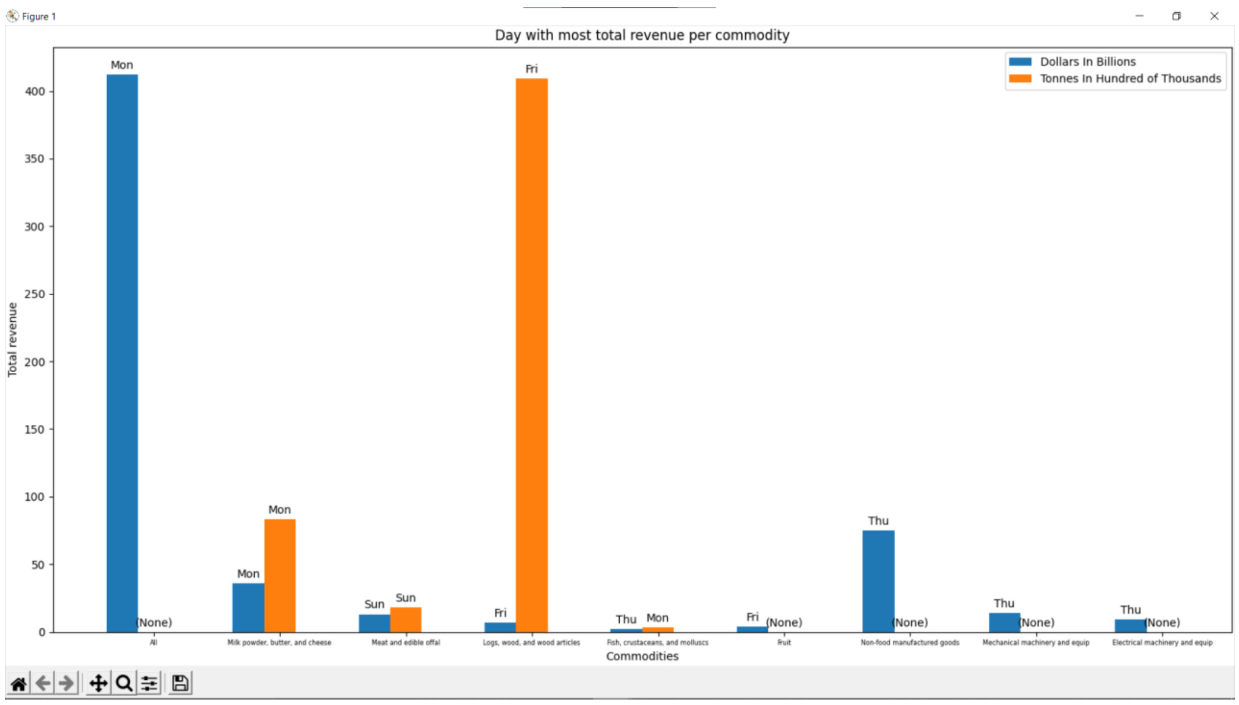


Figure 1
Day with most total revenue per commodity
Dollars In Billions
Tonnes In Hundred of Thousands
Total revenue
400
350
300
250
200
150
100
50
0
Mon
(None)
Mon
Mon
Sun
Sun
Fri
Fri
Thu
Mon
Fri
(None)
Thu
(None)
Thu
(None)
Thu
(None)
All
Milk powder, butter, and cheese
Meat and edible offal
Logs, wood, and wood articles
Fish, crustaceans, and molluscs
Fruit
Non-food manufactured goods
Mechanical machinery and equip
Electrical machinery and equip
Commodities

<u>Σχόλια - Παραδοχές που έγιναν για την ανάπτυξη της εργασίας</u>

Το αρχείο “mysqlconfig.txt” πρέπει να συμπληρωθεί κατάλληλα πριν την εκτέλεση του κώδικα ώστε να μπορέσει το πρόγραμμα να συνδεθεί στην βάση δεδομένων.

Στα διαγράμματα οι αριθμοί έχουν αποκοπεί ώστε να είναι πιο ευανάγνωστο το διάγραμμα, αλλά έχουν χαθεί κάποια ψηφία. Υπέθεσα ότι είναι καλύτερο να είναι ευανάγνωστο από το να είναι ακριβές. Για τα ακριβή δεδομένα μπορείτε να ανατρέξετε στην βάση δεδομένων ή στα αρχεία csv.΄

Τα παράθυρα των διαγραμμάτων πρέπει να μεγιστοποιούνται ώστε να φανούν καλά τα νούμερα και οι ετικέτες.

Δεν πρόλαβα να προσθέσω όλα τα απαραίτητα σχόλια στον κώδικα.