Εργαστηριακή Άσκηση Python

Όνομα: Φίλιππος-Παρασκευάς Ζυγούρης

ΑΜ: 1084660

Έτος: 3ο

E-mail: up1084660@upnet.gr

ΕΙΣΑΓΩΓΗ

Η ανάλυση των εμπορικών δεδομένων είναι απαραίτητη για την κατανόηση της απόδοσης του εμπορικού κλάδου σε μια συγκεκριμένη περίοδο. Σε αυτήν την αναφορά, αναλύω ένα σύνολο δεδομένων συναλλαγών που περιέχει πληροφορίες σχετικά με το εμπόριο αγαθών μεταξύ διαφορετικών χωρών. Το σύνολο δεδομένων περιέχει πληροφορίες για την κατεύθυνση του εμπορίου, το έτος, την ημερομηνία, την ημέρα της εβδομάδας, τη χώρα, το εμπόρευμα, τον τρόπο μεταφοράς, το μέτρο, την αξία και τις σωρευτικές αξίες. Έχω χρησιμοποιήσει τη γλώσσα προγραμματισμού Python και πολλές βιβλιοθήκες, συμπεριλαμβανομένων των Pandas και Matplotlib για την ανάλυση του συνόλου δεδομένων.

Η εικόνα έχει σχεδιαστεί για να αντιπροσωπεύει την ιδέα ότι ο προγραμματισμός Python μπορεί να βοηθήσει να ξεπεραστούν οι προκλήσεις που θέτει ο COVID-19 μέσω της ανάλυσης δεδομένων.

DATA CLEANING

Το πρώτο βήμα ήταν να φορτωθεί το σύνολο δεδομένων σε ένα Pandas DataFrame. Με αυτόν τον τρόπο διασφαλίζω ότι τα δεδομένα είναι καθαρά και προ επεξεργασμένα. Επιπλέον εξασφαλίζω ότι οι στήλες έχουν τους σωστούς τύπους δεδομένων και όπου ήταν απαραίτητο, έγινε η κατάλληλη μετατροπή.

DATA AGGREGATION

Πραγματοποίησα διάφορες εργασίες συγκέντρωσης δεδομένων για να εξαγάγω τις σημαντικές πληροφορίες από το σύνολο δεδομένων. Συγκέντρωσα τα δεδομένα ανά μήνα, χώρα, μέσο μεταφοράς, κατηγορία αγαθών και ανά ημέρα της εβδομάδας.

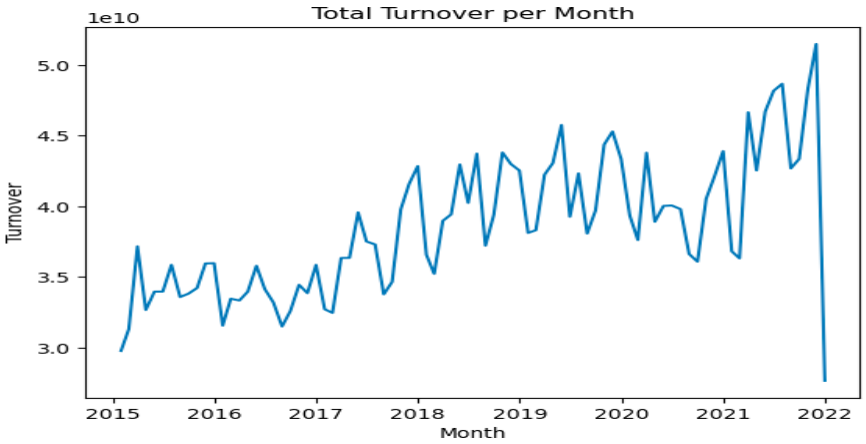
ΑΠΟΤΕΛΈΣΜΑΤΑ ΚΑΙ ΕΡΜΗΝΕΊΑ

1. Υπολόγισα και σχεδίασα τη συνολική παρουσίαση του τζίρου (στήλη αξίας) ανά μήνα. Η γραφική παράσταση έδειξε ότι ο τζίρος ήταν υψηλότερος σε ορισμένους μήνες από άλλους, και ότι υπήρχε μια εποχιακή τάση στον τζίρο των αγαθών.

   Ακολουθεί ένα δείγμα κώδικα για τον υπολογισμό και τη σχεδίαση του συνολικού τζίρου ανά μήνα:

```
# Group by month and sum the values

monthly_turnover = df.groupby(pd.Grouper(key='Date',
freq='M'))['Value'].sum()

monthly_turnover

plt.plot(monthly_turnover.index, monthly_turnover.values)

plt.title('Total Turnover per Month')

plt.xlabel('Month')

plt.ylabel('Turnover')

plt.show()
```

Σε αυτόν τον κώδικα, μετατρέπω πρώτα τη στήλη **Ημερομηνία** σε μορφή datetime χρησιμοποιώντας τη συνάρτηση pd.to_datetime(). Στη συνέχεια, χρησιμοποιώ τη συνάρτηση groupby() για να ομαδοποιήσω τα δεδομένα ανά μήνα (freq='M') και να αθροίσω τις τιμές στη στήλη **Value**. Τέλος, σχεδιάζω τα αποτελέσματα χρησιμοποιώντας το matplotlib.



2. Επίσης, υπολόγισα και σχεδίασα τη συνολική παρουσίαση του τζίρου (στήλη αξίας) για κάθε χώρα. Η γραφική παράσταση έδειξε ότι ορισμένες χώρες είχαν υψηλότερο τζίρο από άλλες και υπήρχε σημαντική διαφορά στους τζίρους μεταξύ των χωρών.

   Ακολουθεί ο κωδικός για τον υπολογισμό και τη σχεδίαση του συνολικού τζίρου ανά χώρα:

```
country_turnover = df.groupby('Country')['Value'].sum()

country_turnover

plt.figure(figsize=(12,5))

plt.bar(country_turnover.index, country_turnover.values)

plt.title('Total Turnover per Country')

plt.xlabel('Country')

plt.ylabel('Turnover')

plt.xticks(rotation=30)

plt.show()
```

Σε αυτόν τον κώδικα, χρησιμοποιώ τη συνάρτηση groupby() για να ομαδοποιήσω τα δεδομένα ανά χώρα και να αθροίσω τις τιμές στη στήλη **Τιμή**. Στη συνέχεια, σχεδιάζω τα αποτελέσματα σε ραβδόγραμμα χρησιμοποιώντας το matplotlib. Ο άξονας x δείχνει τα

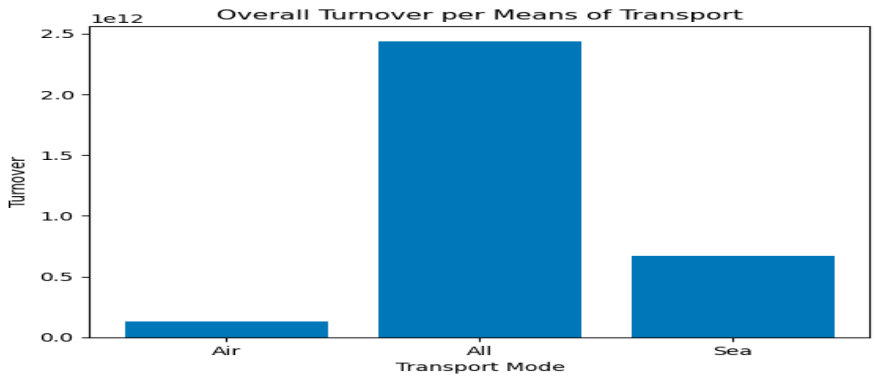
ονόματα των χωρών και ο άξονας y δείχνει τον συνολικό τζίρο.

3. Επιπλέον, υπολόγισα και σχεδίασα τη συνολική παρουσίαση του τζίρου (στήλη αξίας) για κάθε μεταφορικό μέσο. Η γραφική παράσταση έδειξε ότι ορισμένα μεταφορικά μέσα είχαν μεγαλύτερο τζίρο από άλλα και υπήρχε σημαντική διαφορά στους τζίρους μεταξύ των μέσων μεταφοράς.

   Ακολουθεί το δείγμα κώδικα για τον υπολογισμό και τη σχεδίαση του συνολικού τζίρου ανά μέσο μεταφοράς:

```
# Group by transport mode and sum the values

transport_turnover =
df.groupby('Transport_Mode')['Value'].sum()

transport_turnover

plt.bar(transport_turnover.index, transport_turnover.values)

plt.title('Overall Turnover per Means of Transport')

plt.xlabel('Transport Mode')

plt.ylabel('Turnover')

plt.show()
```

Σε αυτόν τον κώδικα, χρησιμοποιώ τη συνάρτηση groupby() για να ομαδοποιήσω τα δεδομένα της στήλης **μέσο μεταφοράς** και να αθροίσω τις τιμές στη στήλη **Τιμή**. Στη συνέχεια, σχεδιάζω το ραβδόγραμμα χρησιμοποιώντας το matplotlib. Ο άξονας x δείχνει τα μέσα μεταφοράς και ο άξονας y τον συνολικό τζίρο.

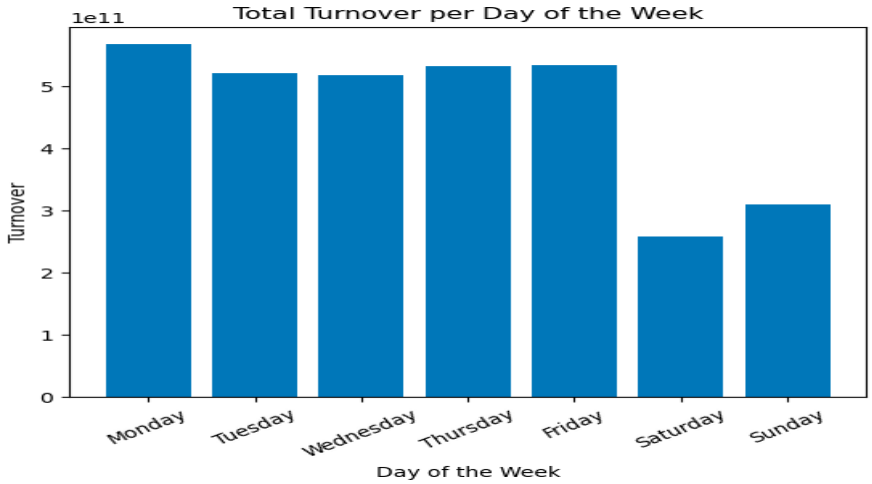


4. Υπολόγισα και σχεδίασα τη συνολική παρουσίαση του τζίρου (στήλη αξίας) για κάθε ημέρα της εβδομάδας. Η γραφική παράσταση έδειξε ότι ορισμένες ημέρες είχαν μεγαλύτερο τζίρο από άλλες και υπήρχε σημαντική διαφορά στους τζίρους μεταξύ των ημερών της εβδομάδας.

   Ακολουθεί το δείγμα κώδικα για τον υπολογισμό και τη σχεδίαση του συνολικού τζίρου ανά ημέρα της εβδομάδας:

```
# Group by weekday and sum the values

weekday_turnover = df.groupby('Weekday')['Value'].sum()


# Define the order of weekdays

weekday_order = ['Monday', 'Tuesday', 'Wednesday',
'Thursday', 'Friday', 'Saturday', 'Sunday']

# Reorder the data by the weekday order

weekday_turnover =
weekday_turnover.reindex(weekday_order)

weekday_turnover

plt.bar(weekday_turnover.index, weekday_turnover.values)

plt.title('Total Turnover per Day of the Week')

plt.xlabel('Day of the Week')

plt.ylabel('Turnover')

plt.xticks(rotation=30)

plt.show()
```

Σε αυτόν τον κώδικα, χρησιμοποιώ την συνάρτηση groupby() για να ομαδοποιήσω τα δεδομένα ανά ημέρα της εβδομάδας και να αθροίσω τις τιμές στη στήλη **Τιμή**. Στη συνέχεια, ορίζω τη σειρά των ημερών χρησιμοποιώντας μια λίστα weekday_order και αναδιατάσσω τα δεδομένα καλώντας τη μέθοδο reindex() στο αντικείμενο ομαδοποιημένων δεδομένων. Τέλος, σχεδιάζω τα αποτελέσματα χρησιμοποιώντας το matplotlib ως ραβδόγραμμα. Ο άξονας x δείχνει τις ημέρες της εβδομάδας και ο άξονας y δείχνει τον συνολικό τζίρο.



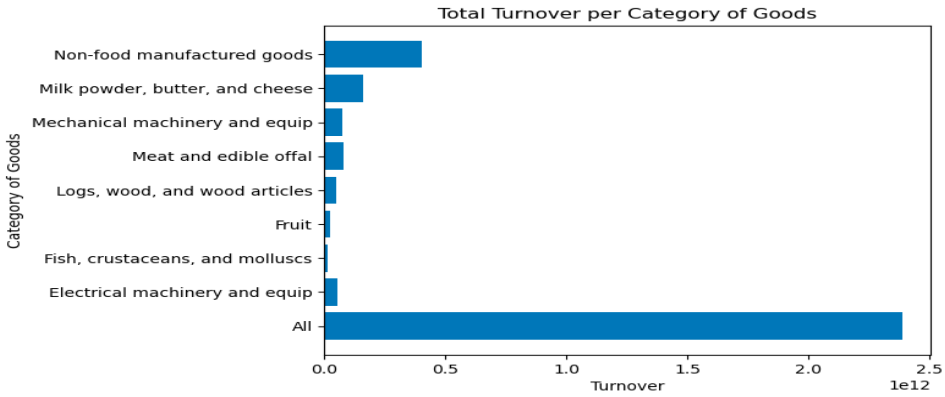
5. Υπολόγισα και σχεδίασα τη συνολική παρουσίαση του τζίρου (στήλη αξίας) για κάθε κατηγορία αγαθών. Η γραφική παράσταση έδειξε ότι ορισμένες κατηγορίες αγαθών είχαν μεγαλύτερο τζίρο από άλλες και υπήρχε σημαντική διαφορά στους τζίρους μεταξύ των κατηγοριών αγαθών.

   Ακολουθεί ο κωδικός για τον υπολογισμό και τη σχεδίαση του συνολικού τζίρου ανά κατηγορία αγαθών:

```
# Group by commodity and sum the values

commodity_turnover =
df.groupby('Commodity')['Value'].sum()

commodity_turnover

plt.barh(commodity_turnover.index,
commodity_turnover.values)

plt.title('Total Turnover per Category of Goods')
```

```
plt.ylabel('Category of Goods')

plt.xlabel('Turnover')

plt.show()
```

Σε αυτόν τον κώδικα, χρησιμοποιώ τη συνάρτηση groupby() για να ομαδοποιήσω τα δεδομένα ανά κατηγορία αγαθών (δηλαδή, εμπόρευμα) και να αθροίσω τις τιμές στη στήλη **Αξία/Τιμή**. Στη συνέχεια, σχεδιάζω τα αποτελέσματα χρησιμοποιώντας το matplotlib ως ραβδόγραμμα. Ο άξονας x δείχνει τις κατηγορίες των αγαθών και ο άξονας y δείχνει τον συνολικό τζίρο.



6. Υπολόγισα και σχεδίασα τους 5 μήνες με τον μεγαλύτερο τζίρο, ανεξάρτητα από το μεταφορικό μέσο και το είδος των ανακυκλώσιμων ειδών.

Ακολουθεί ο κωδικός για τον υπολογισμό και τη σχεδίαση των 5 μηνών με τον υψηλότερο τζίρο:

```
# Group by month and sum the values

# Sort the data by descending order and get the top 5 months

monthly_turnover = df.groupby(pd.Grouper(key='Date',
freq='M'))['Value'].sum()

top_months =
monthly_turnover.sort_values(ascending=False)[:5]

top_months
```

```
plt.bar(top_months.index.strftime('%B %Y'),
top_months.values)

plt.title('5 Months with the Highest Turnover')

plt.xlabel('Month')

plt.ylabel('Turnover')

plt.xticks(rotation=30)

plt.show()
```

Σε αυτόν τον κώδικα, χρησιμοποιώ τη συνάρτηση groupby() για να ομαδοποιήσω τα δεδομένα ανά μήνα και να αθροίσω τις τιμές στη στήλη **Τιμή**. Στη συνέχεια, ταξινομώ τα δεδομένα με φθίνουσα σειρά χρησιμοποιώντας τη συνάρτηση sort_values() και παίρνω τους κορυφαίους 5 μήνες με τον εντοπισμό των πρώτων 5 τιμών των ταξινομημένων δεδομένων. Τέλος, σχεδιάζω τα αποτελέσματα χρησιμοποιώντας το matplotlib ως ραβδόγραμμα. Ο άξονας x δείχνει τους κορυφαίους 5 μήνες και ο άξονας y δείχνει τον συνολικό τζίρο. Μορφοποιώ τις ετικέτες του άξονα x χρησιμοποιώντας τη μέθοδο strftime() για να εμφανίσω το όνομα του μήνα και το έτος.

7. Υπολόγισα και σχεδίασα την παρουσίαση των 5 κατηγοριών αγαθών με τον μεγαλύτερο τζίρο, για κάθε χώρα.

   Ακολουθεί ο κωδικός για τον υπολογισμό και τη σχεδίαση των 5 κατηγοριών αγαθών με τον μεγαλύτερο κύκλο εργασιών, για κάθε χώρα:

```
# Group by country and commodity, and sum the values

country_commodity_turnover = df.groupby(['Country',
'Commodity'])['Value'].sum()

# Loop over the unique countries in the data

for country in df['Country'].unique():

# Get the top 5 commodities for the current country

top_commodities =
country_commodity_turnover.loc[country].nlargest(5)
```

```
# Plot the results for the current country
plt.figure(figsize=(9,4))
plt.bar(top_commodities.index, top_commodities.values)
plt.title(f'Top 5 Commodities with the Largest Turnover in {country}')
plt.xlabel('Commodity')
plt.ylabel('Turnover')
plt.xticks(rotation=30)
plt.show()
```

Σε αυτόν τον κώδικα, χρησιμοποιώ τη συνάρτηση groupby() για να ομαδοποιήσω τα δεδομένα ανά χώρα και εμπόρευμα και να αθροίσω τις τιμές στη στήλη **Αξία**. Στη συνέχεια, εκτελώ τον βρόγχο πάνω από τις μοναδικές χώρες στα δεδομένα και για κάθε χώρα, παίρνω τα 5 κορυφαία εμπορεύματα με τον μεγαλύτερο τζίρο, εντοπίζοντας τις πρώτες 5 τιμές των ομαδοποιημένων δεδομένων χρησιμοποιώντας τη μέθοδο nlargest(). Τέλος, σχεδιάζω τα αποτελέσματα για κάθε χώρα χρησιμοποιώντας το matplotlib ως ραβδόγραμμα. Ο άξονας x δείχνει τα 5 κορυφαία εμπορεύματα και ο άξονας y δείχνει τον συνολικό τζίρο.

8. Υπολόγισα και σχεδίασα το γράφημα της ημέρας με τον μεγαλύτερο τζίρο, για κάθε κατηγορία εμπορευμάτων.

   Για να υπολογίσω και να σχεδιάσω την παρουσίαση της ημέρας με τον υψηλότερο τζίρο για κάθε κατηγορία εμπορευμάτων, ακολούθησα τα εξής βήματα:

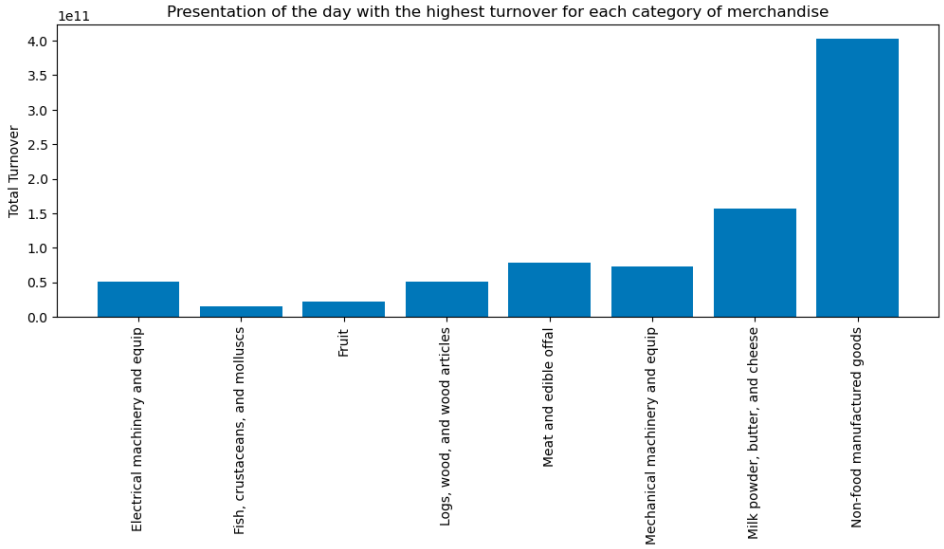
   Φίλτραρα το πλαίσιο δεδομένων για να περιλαμβάνει μόνο σειρές όπου το εμπόρευμα δεν είναι "Όλα".
   Ομαδοποιώ το πλαίσιο δεδομένων που προκύπτει κατά τη στήλη του εμπορεύματος.
   Για κάθε ομάδα, βρίσκω την ημέρα με τον υψηλότερο τζίρο ομαδοποιώντας τα δεδομένα με βάση τη στήλη ημερομηνία και αθροίζοντας τις τιμές για κάθε ημέρα. Στη συνέχεια, ταξινομώ το πλαίσιο δεδομένων που προκύπτει κατά το άθροισμα των τιμών σε φθίνουσα σειρά και επιλέγω τη γραμμή με το υψηλότερο άθροισμα.

Σχεδιάζω το πλαίσιο δεδομένων που προκύπτει χρησιμοποιώντας ένα γράφημα ράβδων, όπου ο άξονας x είναι το εμπόρευμα και ο άξονας y είναι το ποσοστό του συνολικού τζίρου που πραγματοποιήθηκε την ημέρα με τον υψηλότερο τζίρο.
Παρακάτω εμφανίζω τον κώδικα:

```
# filter out rows where the commodity is "All"

df = df[df['Commodity'] != 'All']



# group the data by commodity

groups = df.groupby('Commodity')

# create an empty dataframe to hold the results

results = pd.DataFrame(columns=['Commodity', 'Day',
'Turnover'])

# loop through each group and find the day with the highest
turnover

for name, group in groups:

# group the data by date and sum the values for each day

by_date = group.groupby('Date')['Value'].sum()

# sort the resulting dataframe by sum of values in descending
order

sorted_by_date = by_date.sort_values(ascending=False)

# select the row with the highest sum

highest_day = sorted_by_date.iloc[0]

# calculate the percentage of total turnover that occurred on
the highest day

total_turnover = group['Value'].sum()

# add the result to the results dataframe
```

```
results = results.append({'Commodity': name, 'Day':
sorted_by_date.index[0], 'Turnover': total_turnover},
ignore_index=True)

plt.figure(figsize=(12,4))

plt.bar(results['Commodity'], results['Turnover'])

plt.xticks(rotation=90)

plt.ylabel('Total Turnover')

plt.title('Presentation of the day with the highest turnover for
each category of merchandise')

plt.show()
```

Αυτός ο κωδικός παράγει ένα γράφημα ράβδων που δείχνει το ποσοστό του συνολικού τζίρου που πραγματοποιήθηκε την ημέρα με τον υψηλότερο τζίρο για κάθε κατηγορία εμπορευμάτων. Ο άξονας x θα εμφανίζει τις διάφορες κατηγορίες εμπορευμάτων και ο άξονας y θα δείχνει το ποσοστό του συνολικού τζίρου.



ABOUT CODE SCRIPTS

Για την εργασία δημιούργησα τα ακόλουθα scripts :

- **download.py** (Αυτό το sript θα κατεβάσει δεδομένα από τον συγκεκριμένο σύνδεσμο που μας δόθηκε στην εργασία και θα τα αποθηκεύσει ως αρχείο csv.)

- **Analysis.ipynb** (Αυτό είναι το αρχείο Jupyter που φορτώνει τα δεδομένα από το αρχείο csv στο πλαίσιο δεδομένων, καθαρίζει τα δεδομένα ,τα αναλύει και τα οπτικοποιεί σύμφωνα με τις οδηγίες που δίνονται στην περιγραφή της εργασίας.)

- **MySQL.py** (Αυτό το αρχείο θα συνδεθεί στην κενή βάση δεδομένων μιας MySQL και θα δημιουργήσει πίνακες στη βάση δεδομένων. Στη συνέχεια θα φορτώσει τα δεδομένα από το αρχείο csv στη βάση δεδομένων MYSQL και θα τα αποθηκεύσει σε ένα αρχείο csv που ονομάζεται covid90_table.csv.)

- **GUI.py** (Αυτό το αρχείο θα εμφανίσει ένα μενού όπου μπορείτε να επιλέξετε οποιαδήποτε γραφική παράσταση θέλετε να εμφανιστεί σχετικά με τα δεδομένα. Αυτή είναι η οπτικοποίηση που βασίζεται στο GUI όπου η γραφική παράσταση μπορεί να επιλεγεί από το μενού.)

ABOUT CODE FILES

Για την εργασία δημιούργησα τα ακόλουθα αρχεία :

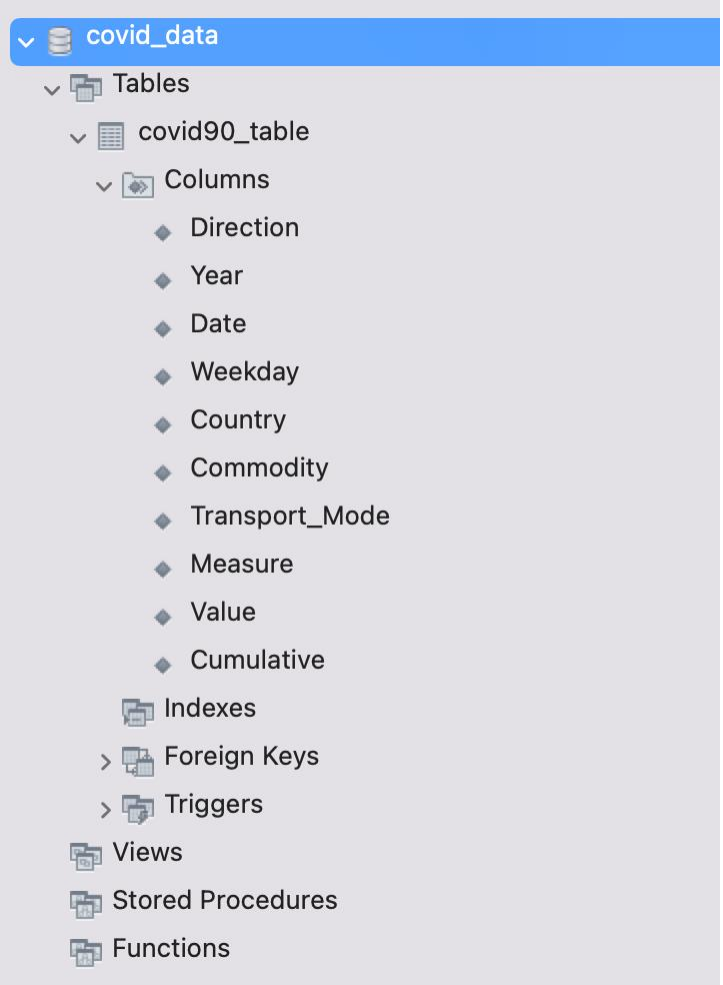
- **1084660_ΖΥΓΟΥΡΗΣ_ΦΙΛΙΠΠΟΣ-ΠΑΡΑΣΚΕΥΑΣ.docx** (αυτό το αρχείο είναι η παρούσα αναφορά)

- **covid_trade_data.csv** (Αυτό το αρχείο το κατέβασα από τον σύνδεσμο χρησιμοποιώντας το python script που ονομάζεται download.py.)

- **covid90_table.csv** (αυτό το αρχείο δημιουργήθηκε από τα δεδομένα MySQL. Περιέχει όλα τα δεδομένα που υπάρχουν στον πίνακα της MySQL.)

- **SQL_DATA.sql** (αυτό είναι ένα αρχείο sql που περιέχει όλα τα δεδομένα και τα ερωτήματα για τη βάση δεδομένων)

SCREENSHOTS OF MYSQL



ANALYSIS.IPYNB

In[1]:

 # install the important modules

!pip install pandas

Εγκαθιστώ την βιβλιοθήκη pandas. Το Pandas είναι μια βιβλιοθήκη Python που παρέχει εργαλεία χειρισμού και ανάλυσης δεδομένων. Το θαυμαστικό πριν από το pip χρησιμοποιείται για την εκτέλεση της εντολής στο τερματικό από το Jupyter Notebook.

!pip install matplotlib

Εγκαθιστώ την βιβλιοθήκη matplotlib. Το Matplotlib είναι μια βιβλιοθήκη σχεδίασης για Python και χρησιμοποιείται για τη δημιουργία οπτικοποιήσεων υψηλής ποιότητας στην Python. Όπως και η προηγούμενη εντολή, χρησιμοποιεί το θαυμαστικό πριν από το pip για να εκτελέσει την εντολή στο τερματικό του Jupyter Notebook.

Μόλις εκτελεστούν οι δύο παραπάνω εντολές, οι βιβλιοθήκες pandas και matplotlib εγκαθίστανται στον υπολογιστή και πλέον είναι έτοιμες και μπορούν να χρησιμοποιηθούν στο πρόγραμμά Python.

In[2]:

 # import these modules

import pandas as pd

Εκχωρώ στην pandas το ψευδώνυμο **pd**, ώστε να αναφέρομαι εύκολα σε αυτή σε ολόκληρο τον κώδικα.

import matplotlib.pyplot as plt

Εκχωρώ στο pyplot της βιβλιοθήκης matplotlib το ψευδώνυμο **plt**, ώστε να αναφέρομαι εύκολα σε αυτό σε ολόκληρο τον κώδικα.

In[3]:

 # load the data into the dataframe from a csv file

df = pd.read_csv('covid_trade_data.csv')

Διαβάζω τα δεδομένα από το CSV αρχείο covid_trade_data.csv και δημιουργώ ένα αντικείμενο pandas DataFrame που ονομάζεται **df**. Η συνάρτηση read_csv() παρέχεται από τη βιβλιοθήκη pandas, διαβάζει το αρχείο CSV και το μετατρέπει σε DataFrame, δηλαδή σε μια δισδιάστατη δομή δεδομένων σαν πίνακα που αποτελείται από γραμμές και στήλες.

df.head()

Εμφανίζω μερικές από τις πρώτες σειρές του DataFrame χρησιμοποιώντας τη μέθοδο head(). Η μέθοδος head() χρησιμοποιείται για την προβολή των επάνω n σειρών του DataFrame. Από προεπιλογή (default), εμφανίζω τις πρώτες 5 σειρές του DataFrame. Παρέχω μια γρήγορη ματιά στα δεδομένα που έχουν φορτωθεί στο DataFrame και διασφαλίζω ότι τα δεδομένα έχουν φορτωθεί σωστά.

Out[3]:

	Direction	Year	Date	Weekday	Country	Commodity	Transport_Mode	Measure	Value	Cumulative
0	Exports	2015	01/01/2015	Thursday	All	All	All	$	104000000	104000000
1	Exports	2015	02/01/2015	Friday	All	All	All	$	96000000	200000000
2	Exports	2015	03/01/2015	Saturday	All	All	All	$	61000000	262000000
3	Exports	2015	04/01/2015	Sunday	All	All	All	$	74000000	336000000
4	Exports	2015	05/01/2015	Monday	All	All	All	$	105000000	442000000

- ✓ **Direction:** Αυτή η στήλη αντιπροσωπεύει την κατεύθυνση του εμπορίου, δηλαδή τα δεδομένα του DataFrame, που σχετίζονται με τις εξαγωγές.
- ✓ **Year:** Αυτή η στήλη αντιπροσωπεύει το έτος κατά το οποίο πραγματοποιήθηκε η συναλλαγή. Για παράδειγμα, το 2015 υποδεικνύει ότι η συναλλαγή πραγματοποιήθηκε το έτος 2015.
- ✓ **Date:** Αυτή η στήλη αντιπροσωπεύει την ημερομηνία της συναλλαγής με τη μορφή HH/MM/EEEE.
- ✓ **Weekday:** Αυτή η στήλη αντιπροσωπεύει την ημέρα της εβδομάδας κατά την οποία έγινε η συναλλαγή.
- ✓ **Country:** Αυτή η στήλη αντιπροσωπεύει τη χώρα που συμμετέχει στο εμπόριο. Το Όλα υποδηλώνει ότι το εμπόριο αφορούσε όλες τις χώρες.
- ✓ **Commodity:** Αυτή η στήλη αντιπροσωπεύει το εμπόρευμα που μεταφέρθηκε. Το Όλα υποδεικνύει ότι όλοι οι τύποι εμπορευμάτων περιλαμβάνονται σε αυτό το σύνολο δεδομένων.
- ✓ **Transport_Mode:** Αυτή η στήλη αντιπροσωπεύει τον τρόπο μεταφοράς που χρησιμοποιήθηκε για την μεταφορά του εμπορίου, όπως Sea, Air κτλπ.
- ✓ **Measure:** Αυτή η στήλη αντιπροσωπεύει τη μονάδα μέτρησης που χρησιμοποιείται για την μέτρηση της αξίας της συναλλαγής. Σε αυτήν την περίπτωση, την μετράμε σε USD ($).
- ✓ **Value:** Αυτή η στήλη αντιπροσωπεύει την αξία της συναλλαγής σε USD ($).
- ✓ **Cumulative:** Αυτή η στήλη αντιπροσωπεύει τη συνολική αξία της συναλλαγής μέχρι αυτό το σημείο.

Explore the data

In[4]:

print("Shape of dataframe:", df.shape)

```
Shape of dataframe: (111438, 10)
```

Με την χρήση της συνάρτησης print() εμφανίζω στην οθόνη το μήνυμα Shape of dataframe:, ακολουθούμενο από την πλειάδα που περιέχει τον αριθμό των γραμμών και στηλών του DataFrame (df.shape - χαρακτηριστικό του αντικειμένου DataFrame του panda που ονομάζεται df , το πρώτο στοιχείο της πλειάδας αντιπροσωπεύει τον αριθμό των σειρών και το δεύτερο στοιχείο τον αριθμό των στηλών) .

Προκύπτει ότι το **DataFrame** έχει 111438 γραμμές και 10 στήλες. Αυτή η πληροφορία είναι απαραίτητη για την κατανόηση του μεγέθους του **DataFrame** και των διαστάσεων του, κάτι που είναι σημαντικό για την εκτέλεση εργασιών ανάλυσης και χειρισμού δεδομένων.

In[5]:

print("Data types of each column:")

print(df.dtypes)

Χρήση της συνάρτησης print(), ώστε να εκτυπώσω τους τύπους δεδομένων κάθε στήλης στο pandas DataFrame – df. Το χαρακτηριστικό dtypes του αντικειμένου DataFrame επιστρέφει ένα αντικείμενο Series που περιέχει τους τύπους δεδομένων κάθε στήλης στο DataFrame.

```
Data types of each column:
Direction          object
Year                int64
Date               object
Weekday            object
Country            object
Commodity          object
Transport_Mode     object
Measure            object
Value               int64
Cumulative          int64
dtype: object
```

- ✓ **Direction object:** Αυτό υποδηλώνει ότι η στήλη Direction του DataFrame περιέχει δεδομένα συμβολοσειράς ή κειμένου, τα οποία αναπαρίστανται ως αντικείμενο στα panda.
- ✓ **Year int64:** Αυτό υποδηλώνει ότι η στήλη Year του DataFrame περιέχει ακέραια δεδομένα, τα οποία αντιπροσωπεύονται ως ακέραιος αριθμός 64 bit.
- ✓ **Date object:** Αυτό υποδηλώνει ότι η στήλη Date object του DataFrame περιέχει δεδομένα συμβολοσειράς ή κειμένου, τα οποία αναπαρίστανται ως αντικείμενο στα panda.
- ✓ **Weekday object :** Αυτό υποδηλώνει ότι η στήλη Weekday του DataFrame περιέχει δεδομένα συμβολοσειράς ή κειμένου, τα οποία αναπαρίστανται ως αντικείμενο στα panda.
- ✓ **Country object :** Αυτό υποδηλώνει ότι η στήλη Country του DataFrame περιέχει δεδομένα συμβολοσειράς ή κειμένου, τα οποία αναπαρίστανται ως αντικείμενο στα panda.
- ✓ **Commodity object:** Αυτό υποδηλώνει ότι η στήλη Commodity του DataFrame περιέχει δεδομένα συμβολοσειράς ή κειμένου, τα οποία αναπαρίστανται ως αντικείμενο στα panda.
- ✓ **Transport_Mode object:** Αυτό υποδηλώνει ότι η στήλη Transport_Mode του DataFrame περιέχει δεδομένα συμβολοσειράς ή κειμένου, τα οποία αναπαρίστανται ως αντικείμενο στα panda.
- ✓ **Measure object:** Αυτό υποδηλώνει ότι η στήλη Measure του DataFrame περιέχει δεδομένα συμβολοσειράς ή κειμένου, τα οποία αναπαρίστανται ως αντικείμενο στα panda.
- ✓ **Value int64:** Αυτό υποδηλώνει ότι η στήλη Value του DataFrame περιέχει ακέραια δεδομένα, τα οποία αντιπροσωπεύονται ως ακέραιος αριθμός 64 bit.
- ✓ **Cumulative int64:** Αυτό υποδηλώνει ότι η στήλη Cumulative του DataFrame περιέχει ακέραια δεδομένα, τα οποία αντιπροσωπεύονται ως ακέραιος αριθμός 64 bit.

Η κατανόηση των τύπων δεδομένων κάθε στήλης είναι σημαντική γιατί πληροφορεί πώς μπορούμε να χειριστούμε τα δεδομένα στο DataFrame. Για παράδειγμα, μπορεί να χρειαστεί να μετατρέψουμε στήλες με δεδομένα συμβολοσειράς σε τύπους αριθμητικών δεδομένων πριν από την εκτέλεση υπολογισμών.

In[6]:

```
print("Number of missing values in each column:")
print(df.isna().sum())
```

Χρήση της συνάρτησης print() για να εμφανιστεί ένα μήνυμα που είναι ο αριθμός των τιμών που λείπουν σε κάθε στήλη του DataFrame. Χρησιμοποιώ τη μέθοδο isna() του DataFrame για να δημιουργήσω ένα νέο DataFrame ίδιου σχήματος με το df, όπου κάθε κελί περιέχει True αν το αντίστοιχο κελί στο df είναι NaN και False στην αντίθετη περίπτωση. Στη συνέχεια, η μέθοδος sum() χρησιμοποιείται σε αυτό το νέο DataFrame για τον υπολογισμό του αριθμού των τιμών True σε κάθε στήλη, που αντιστοιχεί στον αριθμό των τιμών που λείπουν σε κάθε στήλη.

```
Number of missing values in each column:
Direction          0
Year               0
Date               0
Weekday            0
Country            0
Commodity          0
Transport_Mode     0
Measure            0
Value              0
Cumulative         0
dtype: int64
```

Αυτές οι πληροφορίες είναι χρήσιμες ,γιατί βοηθούν στον εντοπισμό τιμών που λείπουν από τις στήλες , οι οποίες πρέπει να χειριστούν κατάλληλα πριν από την περαιτέρω ανάλυση, για την αποφυγή τυχόν προβλημάτων.

In[7]:

print("Number of unique values in each column:")

for col in df.columns:

print(col, ":", df[col].nunique())

Επιτυγχάνεται η εκτύπωση του αριθμού των μοναδικών τιμών σε κάθε στήλη του Panda DataFrame – df. Γίνεται χρήση του βρόγχου for για επανάληψη της κάθε στήλης , ώστε να εμφανίσει το όνομα της τρέχουσας στήλης (col) και τον αριθμό των μοναδικών τιμών σε αυτήν τη στήλη, η οποία λαμβάνεται χρησιμοποιώντας την .nunique( ) μέθοδο του DataFrame.

```
Number of unique values in each column:
Direction : 3
Year : 7
Date : 2541
Weekday : 7
Country : 9
Commodity : 9
Transport_Mode : 3
Measure : 2
Value : 608
Cumulative : 27561
```

Αυτές οι πληροφορίες είναι χρήσιμες για την κατανόηση της δομής των δεδομένων και τον προσδιορισμό στηλών με μικρό αριθμό μοναδικών τιμών, οι οποίες μπορεί να μην είναι κατατοπιστικές για ανάλυση. Μπορεί επίσης να βοηθήσει στον εντοπισμό στηλών με μεγάλο αριθμό μοναδικών τιμών, οι οποίες μπορεί να απαιτούν ειδικό χειρισμό σε ορισμένους τύπους ανάλυσης.

1).Συνολική παρουσίαση του τζίρου (στήλη value) ανά μήνα (στις αντίστοιχες μονάδες μέτρησης)

In[8]:

 # Convert date column to datetime format

df['Date'] = pd.to_datetime(df['Date'], format='%d/%m/%Y')

Χρήση της συνάρτησης pd.to_datetime() για τη μετατροπή της στήλης ημερομηνίας του το DataFrame σε μορφή ημερομηνίας. Η παράμετρος format καθορίζει τη μορφή των ημερομηνιών στη στήλη.

 # Group by month and sum the values

monthly_turnover = df.groupby(pd.Grouper(key='Date', freq='M'))['Value'].sum()

monthly_turnover

Ομαδοποιώ τα δεδομένα στο DataFrame ανά μήνα χρησιμοποιώντας το groupby() μέθοδος και την συνάρτηση pd.Grouper(). Η παράμετρος κλειδί καθορίζει τη στήλη που θα ομαδοποιηθεί **(Ημερομηνία**) και η παράμετρος συχνότητα καθορίζει τη συχνότητα των ομάδων (M για μήνες). Η **Τιμή** του καθορίζει τη στήλη που θα συγκεντρωθεί το άθροισμα για κάθε ομάδα. Τέλος, χρησιμοποιώ τη μέθοδος .sum() για τον υπολογισμό του αθροίσματος των τιμών για κάθε ομάδα. Δηλαδή ομαδοποιώ τα δεδομένα ανά μήνα και αθροίζω τις τιμές για κάθε μήνα.

Η έξοδος αυτού του κώδικα είναι ένα νέο DataFrame που ονομάζεται monthly_turnover που περιέχει τα δεδομένα μηνιαίου τζίρου. Το index αυτού του DataFrame είναι ένα αντικείμενο ημερομηνίας που αντιπροσωπεύει την τελευταία ημέρα κάθε μήνα και οι τιμές είναι ο συνολικός τζίρος για κάθε μήνα.

Out[8]:

```
Date
2015-01-31    29765357000
2015-02-28    31316267000
2015-03-31    37169485000
2015-04-30    32665483000
2015-05-31    33963877000
```

```
                ...
2021-08-31    42686040000
2021-09-30    43359969000
2021-10-31    48281505000
2021-11-30    51493153000
2021-12-31    27652220000
Freq: M, Name: Value, Length: 84, dtype: int64
```

Η έξοδος δείχνει τον μηνιαίο τζίρο για τα δεδομένα κατά την περίοδο από τον Ιανουάριο του 2015 έως τον Δεκέμβριο του 2021. Για παράδειγμα, τον Ιανουάριο του 2015, ο συνολικός τζίρος ήταν 29.765.357.000 και τον Δεκέμβριο του 2021 ο συνολικός τζίρος ήταν 27.652.220.000.

In[9]:

plt.plot(monthly_turnover.index, monthly_turnover.values)

Σχεδιάζω τις τιμές στη στήλη Value του monthly_turnover DataFrame στον άξονα y και τις αντίστοιχες τιμές ημερομηνίας στον άξονα x. Το monthly_turnover.index επιστρέφει το δείκτη datetime και το monthly_turnover.values τις τιμές στη στήλη Value.

plt.title('Total Turnover per Month')

Ορίζω τον τίτλο σε Συνολικό τζίρο ανά μήνα.

plt.xlabel('Month')

Ορίζω την ετικέτα για τον άξονα x σε Month.

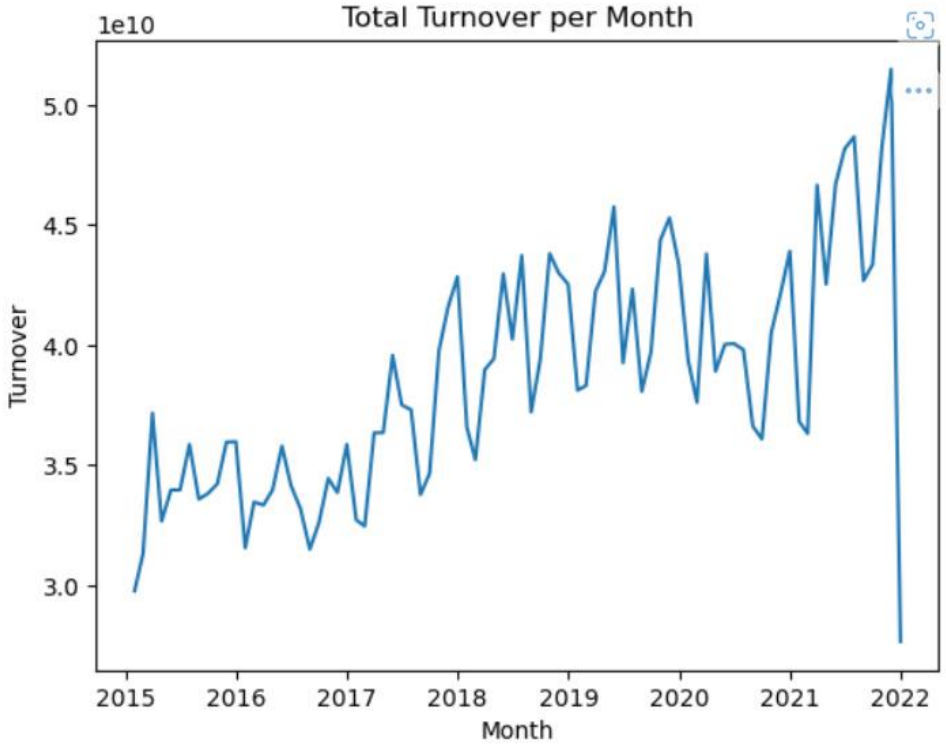
plt.ylabel('Turnover')

Ορίζω την ετικέτα για τον άξονα y σε Turnover.

plt.show()

Εμφανίζω το plot.

Ο κώδικας χρησιμοποιεί τη βιβλιοθήκη matplotlib για να δημιουργήσει την γραφική παράσταση του monthly_turnover DataFrame που δημιουργήθηκε νωρίτερα.



2). Συνολική παρουσίαση του τζίρου (στήλη value) για κάθε χώρα (στις αντίστοιχες μονάδες μέτρησης)

In[10]:

 # Group by country and sum the values

country_turnover = df.groupby('Country')['Value'].sum()

country_turnover

Ο κώδικας ομαδοποιεί τη στήλη **df DataFrame** ανά χώρα και, στη συνέχεια, συγκεντρώνει τη στήλη **Τιμή** χρησιμοποιώντας τη συνάρτηση αθροίσματος. Το αντικείμενο που προκύπτει, country_turnover, είναι Pandas που περιέχει τον συνολικό τζίρο για κάθε χώρα. Κάτι που μπορεί να βοηθήσει στον εντοπισμό των πιο κερδοφόρων χωρών.

Ο κώδικας χρησιμοποιεί τη μέθοδο groupby() του pandas DataFrame για να ομαδοποιήσει τις γραμμές του df DataFrame με τις μοναδικές τιμές στη στήλη Country.

Η συνάρτηση .groupby('Country') του κώδικα δημιουργεί ένα αντικείμενο DataFrameGroupBy.

Με την χρήση του ['Value'].sum() επιλέγω τη στήλη Value από κάθε ομάδα και, στη συνέχεια, υπολογίζω το άθροισμα των τιμών σε αυτήν τη στήλη για κάθε ομάδα.

Το αντικείμενο country_turnover που προκύπτει είναι ένα αντικείμενο της σειράς pandas, όπου εμφανίζονται οι μοναδικές τιμές του Country και το άθροισμα της στήλης Value για κάθε ομάδα.

Out[10]:

```
Country
All                              2315389349000
Australia                         107686000000
China                             282769573000
East Asia (excluding China)       116562137000
European Union (27)                26644000000
Japan                              23155000000
Total (excluding China)           291991000000
United Kingdom                     21591000000
United States                      52321338000
Name: Value, dtype: int64
```

In[11]:

plt.figure(figsize=(12,5))

Ορίζω το μέγεθος του σχήματος που θα δημιουργηθεί, με πλάτος 12 ίντσες και ύψος 5 ίντσες.

plt.bar(country_turnover.index, country_turnover.values)

Δημιουργώ το γράφημα ράβδων, με τις χώρες στον άξονα x και τις αντίστοιχες τιμές τζίρου τους στον άξονα y. Το country_turnover.index επιλέγει τα ονόματα των χωρών ως τιμές του άξονα x και το country_turnover.values επιλέγει τις τιμές συνολικού τζίρου ως τιμές του άξονα y.

plt.title('Total Turnover per Country')

Ορίζω τον τίτλο του γραφήματος σε Συνολικός τζίρος ανά χώρα.

plt.xlabel('Country')

Ορίζω την ετικέτα για τον άξονα x σε Country.

plt.ylabel('Turnover')

Ορίζω την ετικέτα για τον άξονα y σε Turnover.

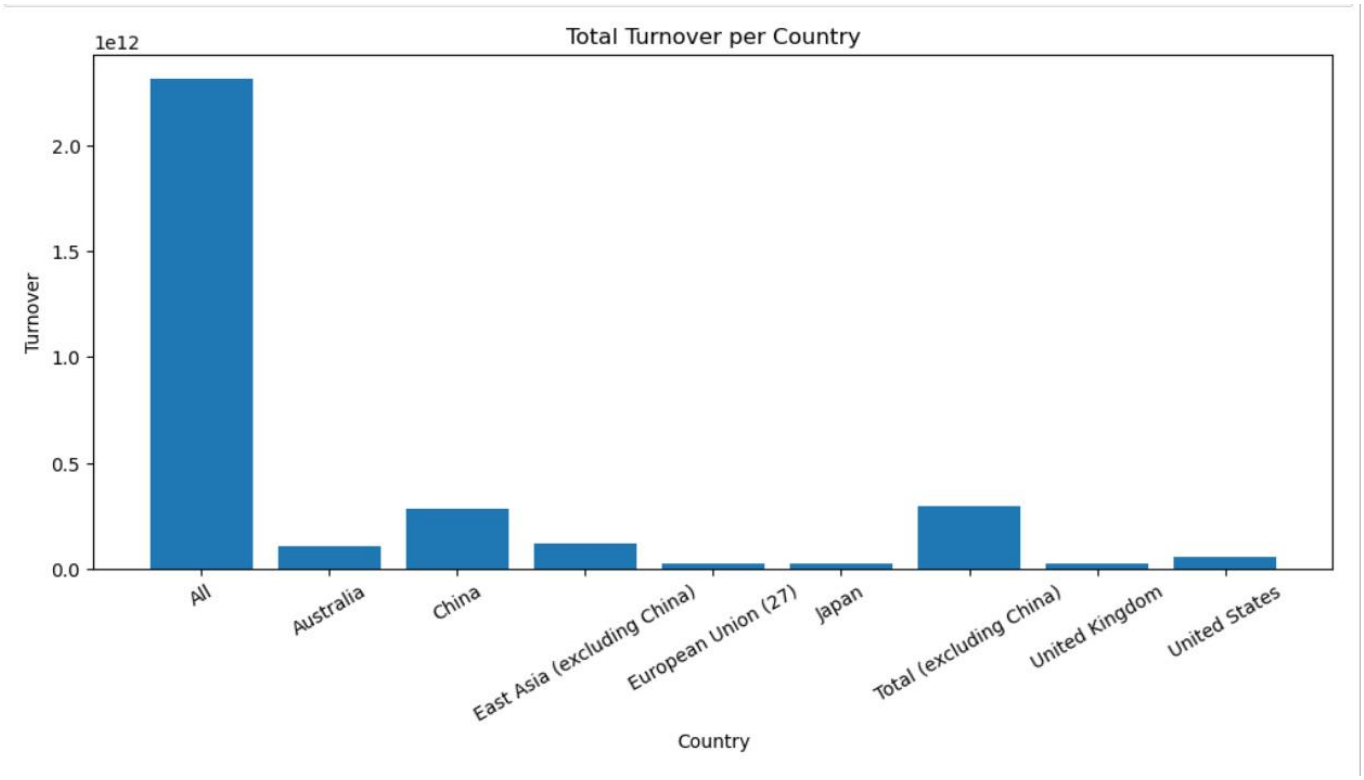
plt.xticks(rotation=30)

Περιστρέφω τις ετικέτες tick του άξονα x κατά 30 μοίρες.

plt.show()

Εμφανίζω το γράφημα.

Δημιουργώ ένα γράφημα ράβδων του συνολικού τζίρου για κάθε χώρα στο σύνολο δεδομένων χρησιμοποιώντας τη βιβλιοθήκη matplotlib.



3). Συνολική παρουσίαση του τζίρου (στήλη value) για κάθε μέσο μεταφοράς (στις αντίστοιχες μονάδες μέτρησης)

In[12]:

 # Group by transport mode and sum the values

transport_turnover = df.groupby('Transport_Mode')['Value'].sum()

transport_turnover

Ομαδοποιώ τις γραμμές του DataFrame - df από τη στήλη Transport_Mode και στη συνέχεια υπολογίζω το άθροισμα της στήλης Τιμή για κάθε ομάδα. Το αποτέλεσμα είναι ένα αντικείμενο Pandas transport_turnover όπου το index αποτελείται από τις μοναδικές τιμές στη στήλη Transport_Mode και οι τιμές αντιπροσωπεύουν τον συνολικό τζίρο για κάθε μέσο μεταφοράς.

Για παράδειγμα, το df περιέχει δεδομένα για αεροπορικές, σιδηροδρομικές, οδικές και θαλάσσιες μεταφορές εμπορευμάτων, οπότε το transport_turnover θα έχει 3 καταχωρήσεις, όπου το index θα είναι ['Air', 'Sea' , 'All'] και οι αντίστοιχες τιμές είναι ο συνολικός τζίρος για κάθε μέσο μεταφοράς.

Out[12]:

```
Transport_Mode
Air      132602000000
All     2437107397000
Sea      668400000000
Name: Value, dtype: int64
```

In[13]:

plt.bar(transport_turnover.index, transport_turnover.values)

Η συνάρτηση plt.bar() δημιουργεί ένα γράφημα ράβδων με τον άξονα x να αντιπροσωπεύει τις μοναδικές τιμές στη στήλη Transport_Mode και τον άξονα y να αντιπροσωπεύει τον συνολικό τζίρο για κάθε μέσο μεταφοράς.

plt.title('Overall Turnover per Means of Transport')

plt.xlabel('Transport Mode')

plt.ylabel('Turnover')

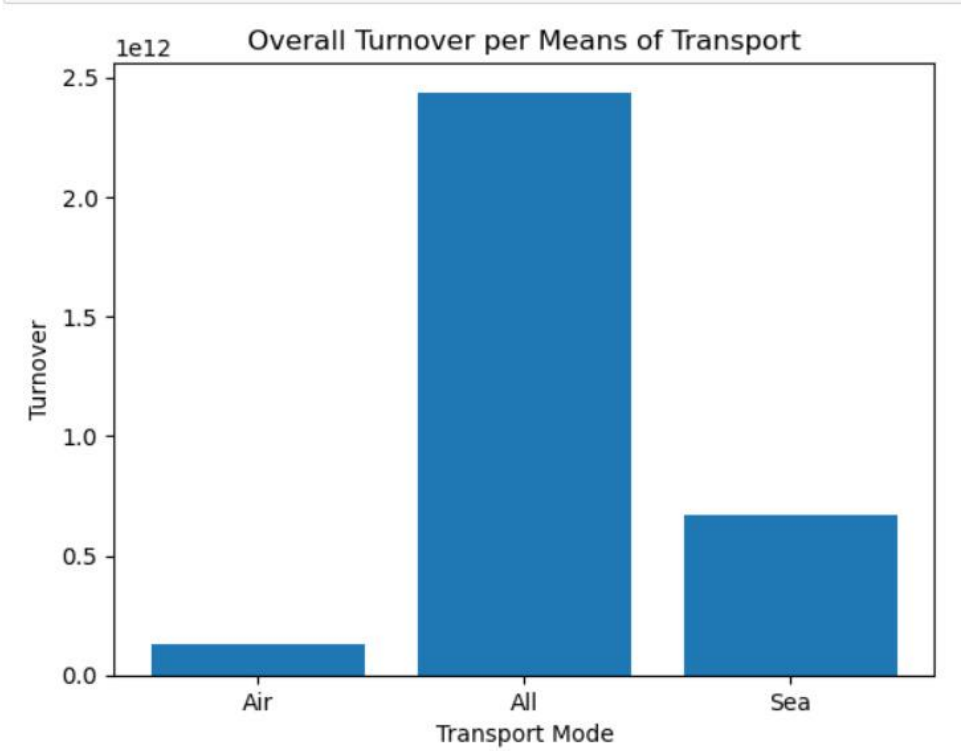
Οι συναρτήσεις plt.title(), plt.xlabel() και plt.ylabel() χρησιμοποιούνται για τον ορισμό του τίτλου, την ετικέτα άξονα x και την ετικέτα του άξονα y του γραφήματος, αντίστοιχα.

plt.show()

Εμφανίζω το γράφημα.

Δημιουργώ ένα γράφημα ράβδων του συνολικού τζίρου για κάθε μέσο μεταφοράς, χρησιμοποιώντας το αντικείμενο της Pandas transport_turnover το οποίο είχε υπολογιστεί προηγουμένως χρησιμοποιώντας τη μέθοδο groupby() για να ομαδοποιήσει τις

γραμμές του dataframe - df στη στήλη Transport_Mode και να υπολογίσει το άθροισμα της στήλης Value για κάθε ομάδα.



4). Συνολική παρουσίαση του τζίρου (στήλη value) για κάθε μέρα της εβδομάδας (στις αντίστοιχες μονάδες μέτρησης)

In[14]:

 # Group by weekday and sum the values

weekday_turnover = df.groupby('Weekday')['Value'].sum()

Ομαδοποιώ τις γραμμές του πλαισίου δεδομένων df κατά τη στήλη Weekday και στη συνέχεια υπολογίζει το άθροισμα της στήλης Value για κάθε ομάδα. Το αποτέλεσμα είναι ένα

αντικείμενο της Pandas weekday_turnover όπου το index αποτελείται από τις μοναδικές τιμές στη στήλη Weekday και οι τιμές αντιπροσωπεύουν τον συνολικό τζίρο για κάθε weekday.

 # Define the order of weekdays

weekday_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']

Ορίζω μια λίστα με τις μέρες της εβδομάδας με την επιθυμητή σειρά.

 # Reorder the data by the weekday order

weekday_turnover = weekday_turnover.reindex(weekday_order)

weekday_turnover

Χρησιμοποιώ τη μέθοδο reindex() για να αναδιατάξω το index του weekday_turnover σύμφωνα με τη σειρά που ορίζεται στο weekday_order. Αυτό διασφαλίζει ότι η προκύπτουσα γραφική παράσταση των δεδομένων θα εμφανίζεται με τη σωστή σειρά των ημερών της εβδομάδας.

Out[14]:

```
Weekday
Monday        566760527000
Tuesday       520511585000
Wednesday     517382998000
Thursday      532329047000
Friday        533137760000
Saturday      258267241000
Sunday        309720239000
Name: Value, dtype: int64
```

In[15]:

plt.bar(weekday_turnover.index, weekday_turnover.values)

Ομαδοποιώ τις γραμμές του πλαισίου δεδομένων df κατά τη στήλη Weekday και υπολογίζω το άθροισμα της στήλης Value για κάθε ομάδα. Το αποτέλεσμα είναι ένα αντικείμενο της

Pandas weekday_turnover όπου το index αποτελείται από τις μοναδικές τιμές των ημερών και οι τιμές αντιπροσωπεύουν τον συνολικό τζίρο για κάθε ημέρα της εβδομάδας.

Ορίζω τη σειρά των ημερών από Δευτέρα έως Κυριακή και αναδιατάσσω τα δεδομένα κατά σειρά ημέρας χρησιμοποιώντας τη μέθοδο reindex() της weekday_turnover. Δημιουργω ένα γράφημα ράβδων των δεδομένων weekday_turnover.

plt.title('Total Turnover per Day of the Week')

plt.xlabel('Day of the Week')

Με τις ημέρες της εβδομάδας ως ετικέτες του άξονα x.

plt.ylabel('Turnover')

Τον συνολικό τζίρο ως ετικέτα του άξονα y.

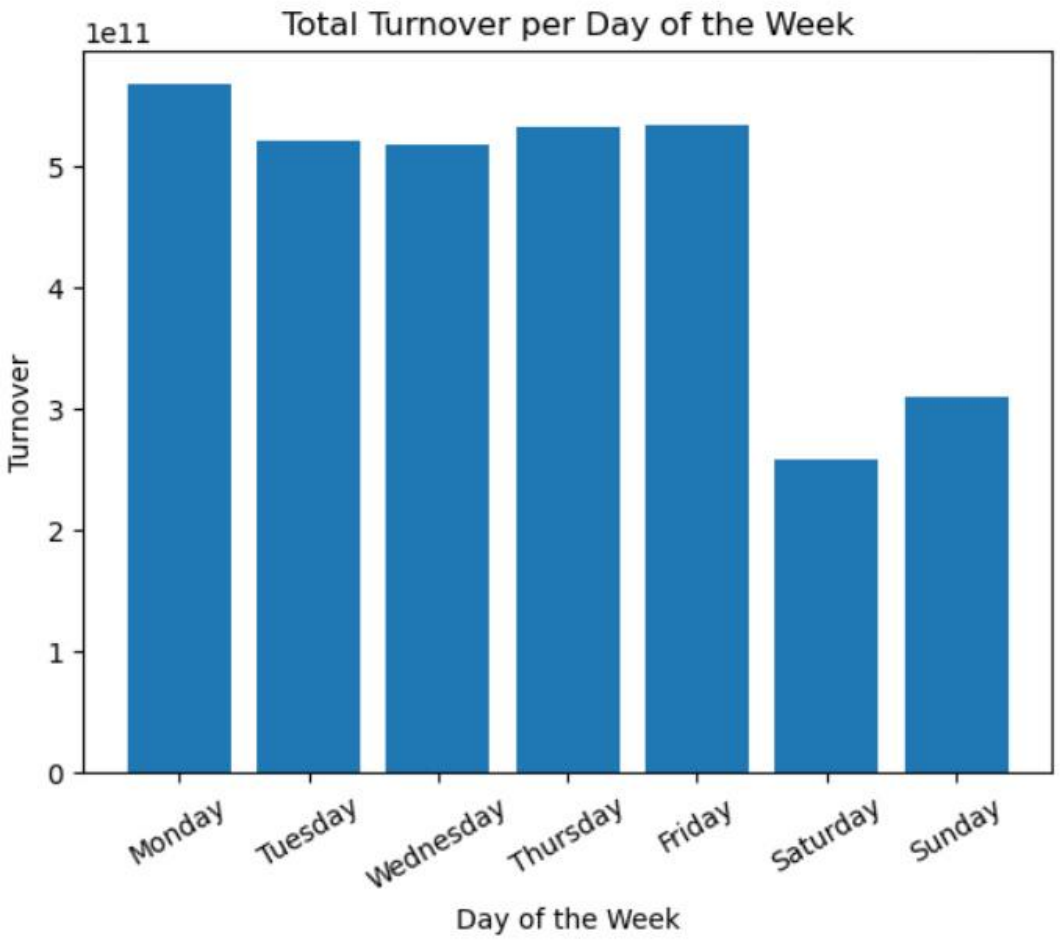
plt.xticks(rotation=30)

Περιστροφή των ετικετών του άξονα x κατά 30 μοίρες για να γίνουν πιο ευανάγνωστες.

plt.show()

Εμφανίζω το γράφημα.

Δημιουργώ ένα γράφημα ράβδων που δείχνει τον συνολικό τζίρο για κάθε ημέρα της εβδομάδας.



5). Συνολική παρουσίαση του τζίρου (στήλη value) για κάθε κατηγορία εμπορεύματος (στις αντίστοιχες μονάδες μέτρησης)

In[16]:

 # Group by commodity and sum the values

commodity_turnover = df.groupby('Commodity')['Value'].sum()

commodity_turnover

Ομαδοποιώ τις γραμμές του DataFrame df κατά τη στήλη Commodity και υπολογίζω το άθροισμα της στήλης Value για

κάθε ομάδα. Το αποτέλεσμα είναι ένα αντικείμενο της Pandas commodity_turnover όπου το index αποτελείται από τις μοναδικές τιμές στη στήλη Commodity και οι τιμές αντιπροσωπεύουν τον συνολικό τζίρο για κάθε εμπόρευμα.

Out[16]:

```
Commodity
All                                2386667000000
Electrical machinery and equip       51554000000
Fish, crustaceans, and molluscs      15446832000
Fruit                                22197000000
Logs, wood, and wood articles        50645402000
Meat and edible offal                78522372000
Mechanical machinery and equip       72603000000
Milk powder, butter, and cheese     157319791000
Non-food manufactured goods         403154000000
Name: Value, dtype: int64
```

In[17]:

plt.barh(commodity_turnover.index, commodity_turnover.values)

Ομαδοποιώ τις γραμμές του πλαισίου δεδομένων df κατά τη στήλη Commodity και στη συνέχεια υπολογίζω το άθροισμα της στήλης Value για κάθε ομάδα. Το αντικείμενο της Pandas που προκύπτει commodity_turnover έχει τις μοναδικές τιμές στη στήλη Commodity και οι τιμές αντιπροσωπεύουν τον συνολικό τζίρο για κάθε κατηγορία αγαθών.

plt.title('Total Turnover per Category of Goods')

Ορίζω τον τίτλο του γραφήματος σε Συνολικό τζίρο ανά κατηγορία εμπορευμάτων

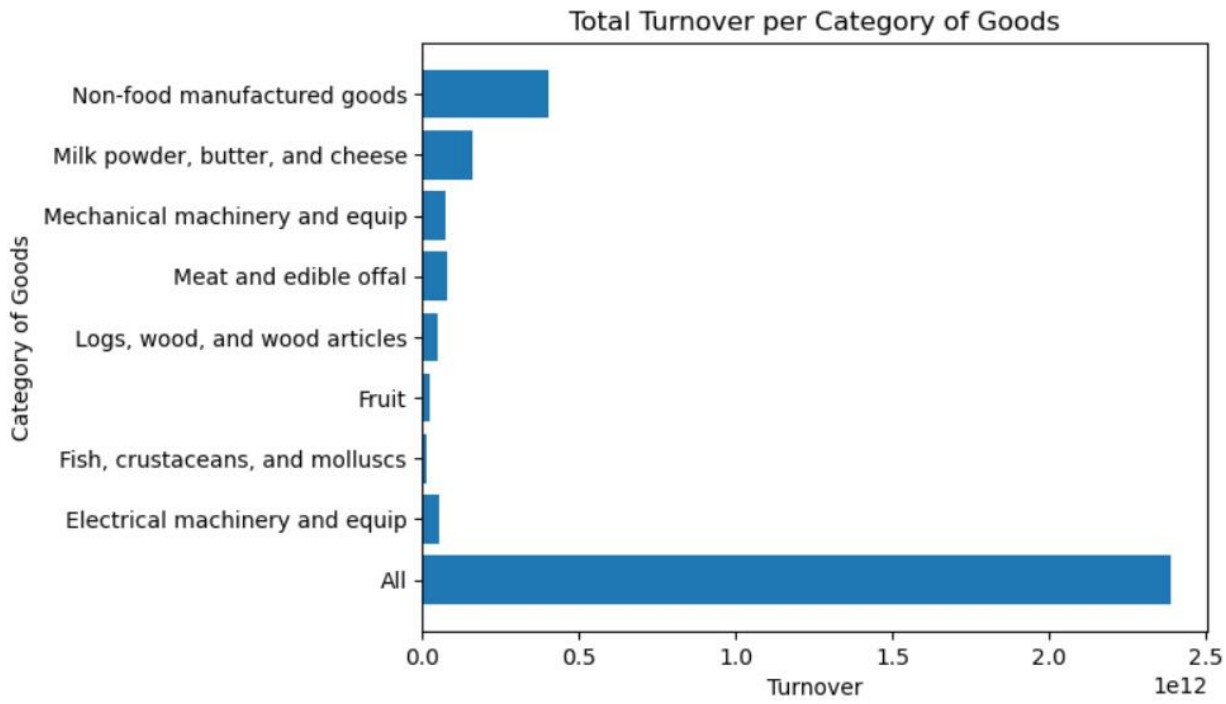
plt.ylabel('Category of Goods')

plt.xlabel('Turnover')

Δημιουργία ενός οριζόντιου ραβδωτού γραφήματος, με την κατηγορία των αγαθών ως άξονα y και τον τζίρο ως τον άξονα x.

plt.show()

Εμφανίζω το γράφημα.



6). Παρουσίαση των 5 μηνών με το μεγαλύτερο τζίρο, ανεξαρτήτως μέσου μεταφοράς και είδους ανακυκλώσιμων ειδών

In[18]:

 # Group by month and sum the values

 # Sort the data by descending order and get the top 5 months

monthly_turnover = df.groupby(pd.Grouper(key='Date', freq='M'))['Value'].sum()

Ομαδοποιώ τις γραμμές του πλαισίου δεδομένων df ανά μήνα και υπολογίζω το άθροισμα της στήλης Value για κάθε μήνα. Το αντικείμενο της Pandas που προκύπτει monthly_turnover έχει τον συνολικό τζίρο για κάθε μήνα.

top_months =
monthly_turnover.sort_values(ascending=False)[:5]

Ταξινομώ τα δεδομένα σε monthly_turnover με φθίνουσα σειρά και επιλέγω τους κορυφαίους 5 μήνες χρησιμοποιώντας τον τελεστή κοπής [:5].

top_months

Το αποτέλεσμα είναι ένα αντικείμενο της Pandas top_months που περιέχει τον συνολικό τζίρο για τους κορυφαίους 5 μήνες με φθίνουσα σειρά.

Out[18]:

```
Date
2021-11-30    51493153000
2021-07-31    48679012000
2021-10-31    48281505000
2021-06-30    48173793000
2021-05-31    46703805000
Name: Value, dtype: int64
```

In[19]:

plt.bar(top_months.index.strftime('%B %Y'),
top_months.values)

Ομαδοποιώ το πλαίσιο δεδομένων df ανά μήνα και υπολογίζω το άθροισμα της στήλης Value για κάθε μήνα χρησιμοποιώντας τη μέθοδο groupby. Το προκύπτον monthly_turnover είναι ένα αντικείμενο της Pandas όπου κάθε index αντιστοιχεί σε έναν μήνα και κάθε τιμή αντιπροσωπεύει τον συνολικό τζίρο για αυτόν τον μήνα.

Στη συνέχεια, ταξινομώ τη σειρά monthly_turnover σε φθίνουσα σειρά και επιλέγω τους κορυφαίους 5 μήνες με τον υψηλότερο τζίρο χρησιμοποιώντας το slicing ([:5]). Οι top_months που προκύπτουν είναι επίσης ένα αντικείμενο της Pandas με την ίδια μορφή με το monthly_turnover.

Δημιουργώ ένα ραβδόγραμμα χρησιμοποιώντας το plt.bar για να απεικονίσει τον συνολικό τζίρο για τους κορυφαίους 5 μήνες.

plt.title('5 Months with the Highest Turnover')

plt.xlabel('Month')

Ο άξονας x δείχνει το όνομα του μήνα και του έτους.

plt.ylabel('Turnover')

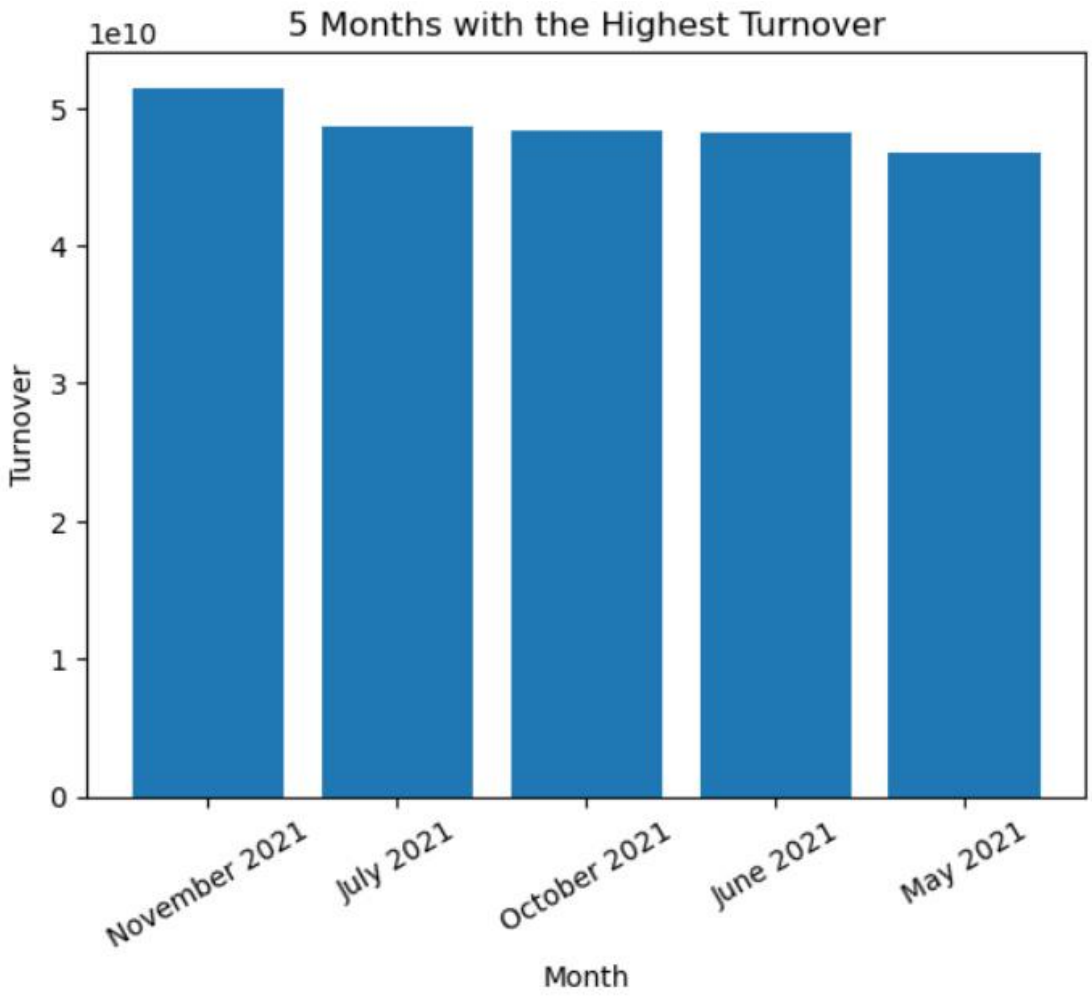
Ο άξονας y δείχνει την αντίστοιχη τιμή τζίρου.

plt.xticks(rotation=30)

Περιστροφή των ετικετών του άξονα x κατά 30 μοίρες για καλύτερη αναγνωσιμότητα.

plt.show()

Εμφάνιση γραφήματος.



7). Παρουσίαση των 5 κατηγοριών εμπορευμάτων με το μεγαλύτερο τζίρο, για κάθε χώρα

In[20]:

```
# Group by country and commodity, and sum the values
country_commodity_turnover = df.groupby(['Country', 'Commodity'])['Value'].sum()
country_commodity_turnover
```

Ομαδοποιώ τις γραμμές του πλαισίου δεδομένων df κατά τις στήλες Country και Commodity και στη συνέχεια υπολογίζω το άθροισμα της στήλης Αξία για κάθε ομάδα. Το αποτέλεσμα είναι ένα αντικείμενο της Pandas country_commodity_turnover όπου ο δείκτης αποτελείται από πλειάδες των μοναδικών τιμών στις στήλες Country και Commodity και οι τιμές αντιπροσωπεύουν τον συνολικό τζίρο για κάθε συνδυασμό χώρας και εμπορεύματος.

Out[20]:

```
Country                    Commodity
All                        All                              1603472000000
                           Electrical machinery and equip     35076000000
                           Fish, crustaceans, and molluscs    11388832000
                           Fruit                              22197000000
                           Logs, wood, and wood articles      32542650000
                           Meat and edible offal              51212749000
                           Mechanical machinery and equip     57567000000
                           Milk powder, butter, and cheese    98779118000
                           Non-food manufactured goods       403154000000
Australia                  All                               107686000000
China                      All                               182406000000
                           Electrical machinery and equip     16478000000
                           Fish, crustaceans, and molluscs     4058000000
                           Logs, wood, and wood articles      18102752000
                           Meat and edible offal              15465285000
                           Mechanical machinery and equip     15036000000
                           Milk powder, butter, and cheese    31223536000
East Asia (excluding China) All                               89245000000
                           Milk powder, butter, and cheese    27317137000
European Union (27)        All                               26644000000
Japan                      All                               23155000000
Total (excluding China)    All                              291991000000
United Kingdom             All                               21591000000
United States              All                               40477000000
                           Meat and edible offal             11844338000
Name: Value, dtype: int64
```

In[21]:

 # Loop over the unique countries in the data

for country in df['Country'].unique():

Δημιουργώ έναν βρόγχο που επαναλαμβάνεται σε κάθε μοναδική χώρα στη στήλη Χώρα του πλαισίου δεδομένων df.

Για κάθε χώρα, επιλέγω τις γραμμές του πλαισίου δεδομένων όπου η στήλη Χώρα αντιστοιχεί στην τρέχουσα χώρα και στη συνέχεια τις ομαδοποιώ με βάση τη στήλη Εμπόρευμα και υπολογίζω το άθροισμα της στήλης Αξία για κάθε εμπόρευμα.

 # Get the top 5 commodities for the current country

top_commodities = country_commodity_turnover.loc[country].nlargest(5)

Στη συνέχεια επιλέγω τα 5 κορυφαία εμπορεύματα με τον μεγαλύτερο τζίρο για την τρέχουσα χώρα χρησιμοποιώντας τη μέθοδο nlargest().

 # Plot the results for the current country

plt.figure(figsize=(9,4))

plt.bar(top_commodities.index, top_commodities.values)

plt.title(f'Top 5 Commodities with the Largest Turnover in {country}')

Ο τίτλος του γραφήματος περιλαμβάνει το όνομα της τρέχουσας χώρας.

plt.xlabel('Commodity')

plt.ylabel('Turnover')

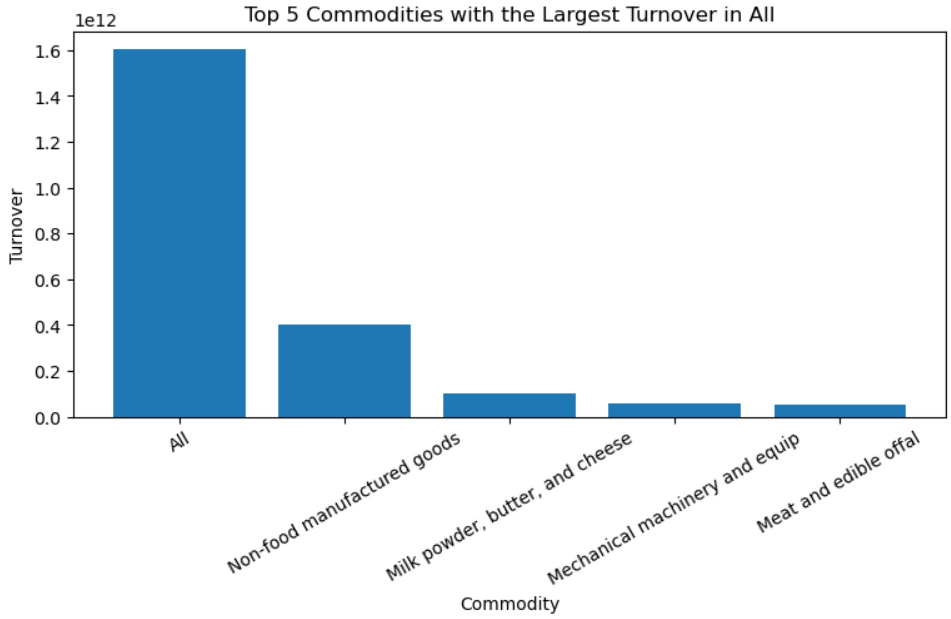
Στη συνέχεια δημιουργώ ένα γράφημα ράβδων για τα 5 κορυφαία εμπορεύματα για την τρέχουσα χώρα, με τα ονόματα των εμπορευμάτων στον άξονα x και τον συνολικό τζίρο στον άξονα y.

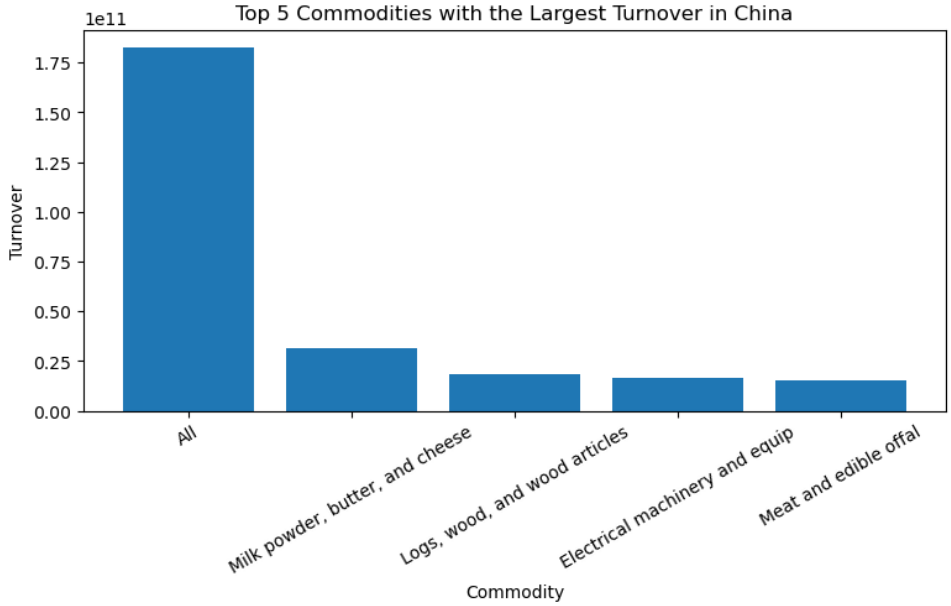
plt.xticks(rotation=30)

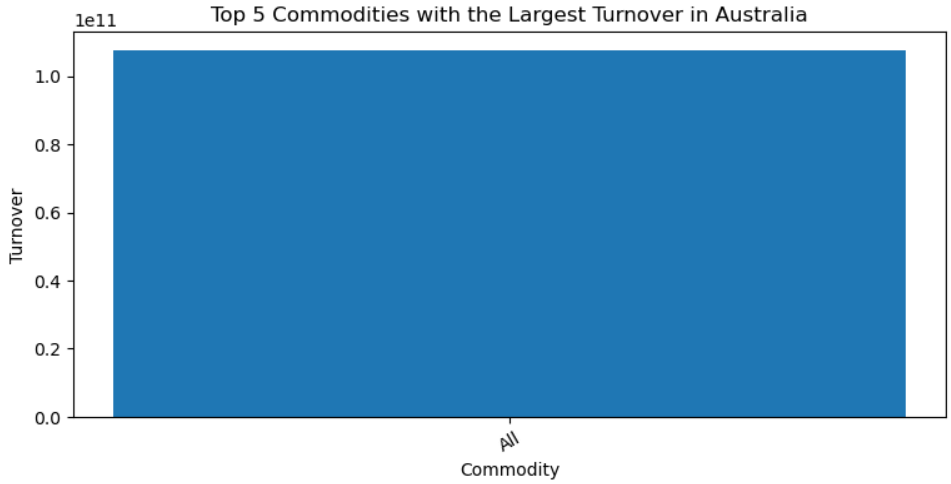
Περιστροφή των ετικετών του άξονα x κατά 30 μοίρες για καλύτερη αναγνωσιμότητα.

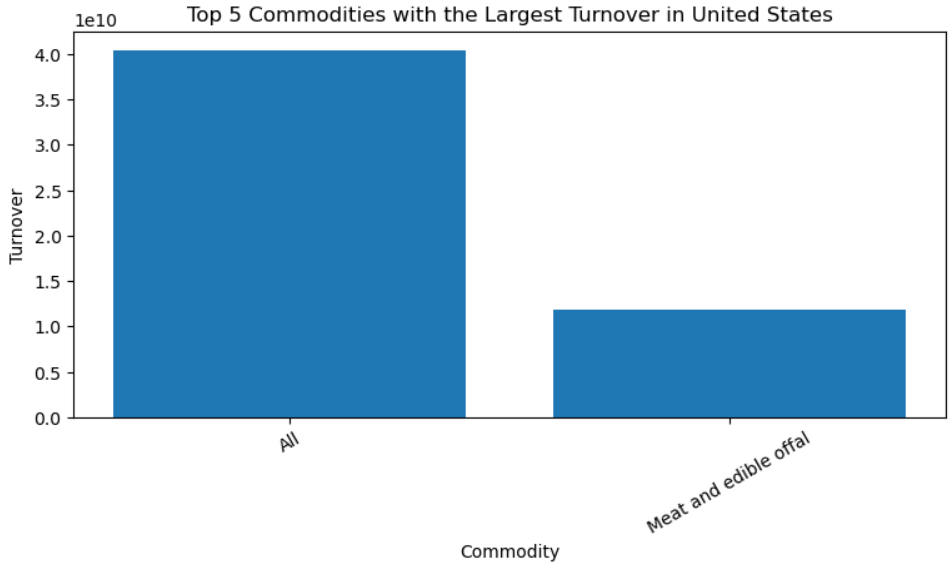
plt.show()

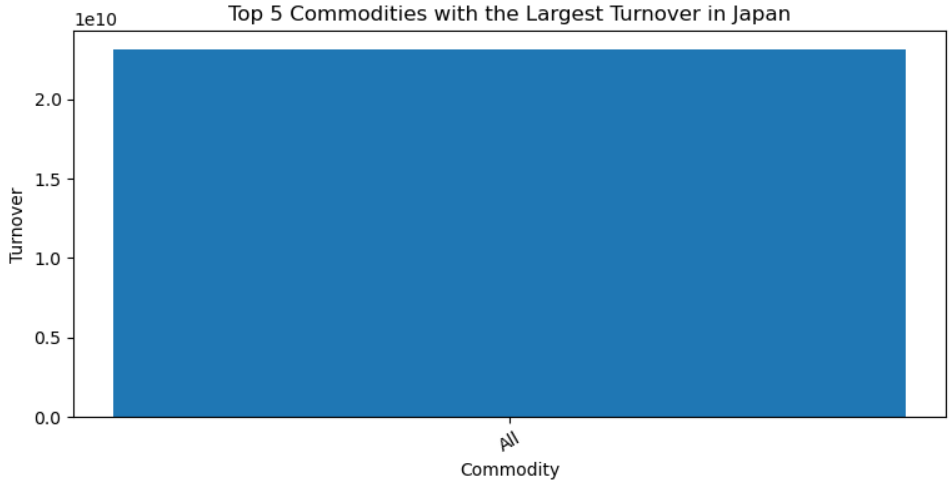
Εμφάνιση του γραφήματος.

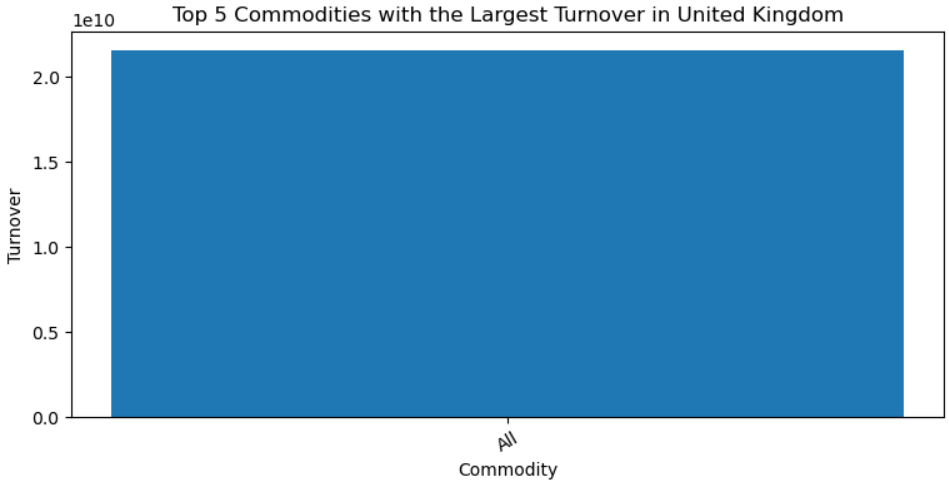

Top 5 Commodities with the Largest Turnover in All
1e12
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Turnover
All
Non-food manufactured goods
Milk powder, butter, and cheese
Mechanical machinery and equip
Meat and edible offal
Commodity

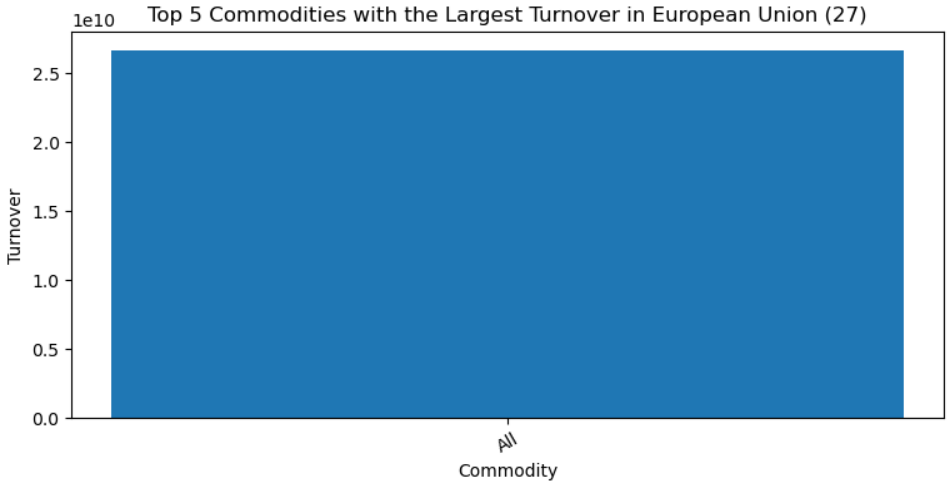

Top 5 Commodities with the Largest Turnover in China
1e11
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Turnover
All
Milk powder, butter, and cheese
Logs, wood, and wood articles
Electrical machinery and equip
Meat and edible offal
Commodity

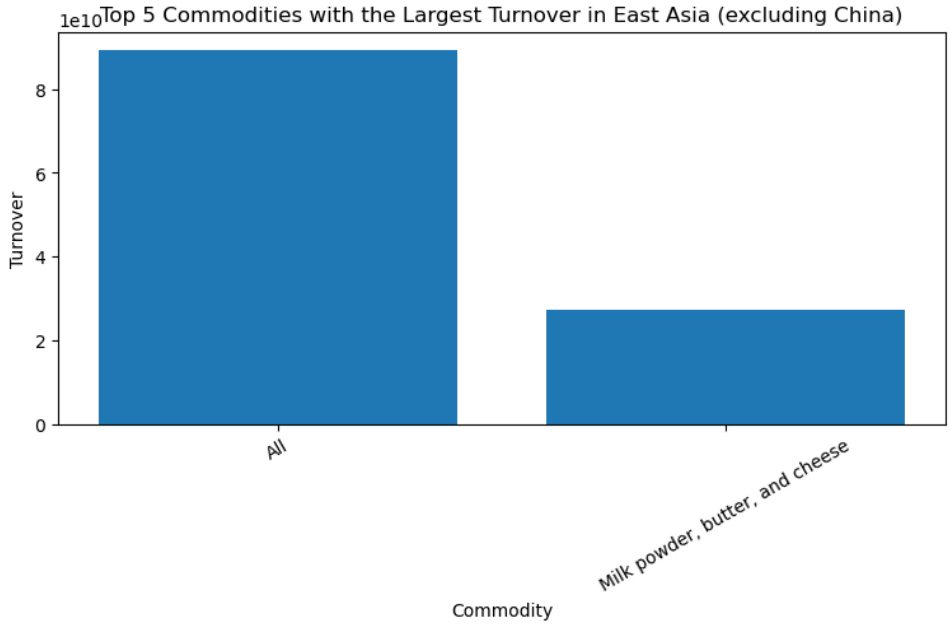

Top 5 Commodities with the Largest Turnover in Australia
1e11
1.0
0.8
0.6
0.4
0.2
0.0
Turnover
All
Commodity

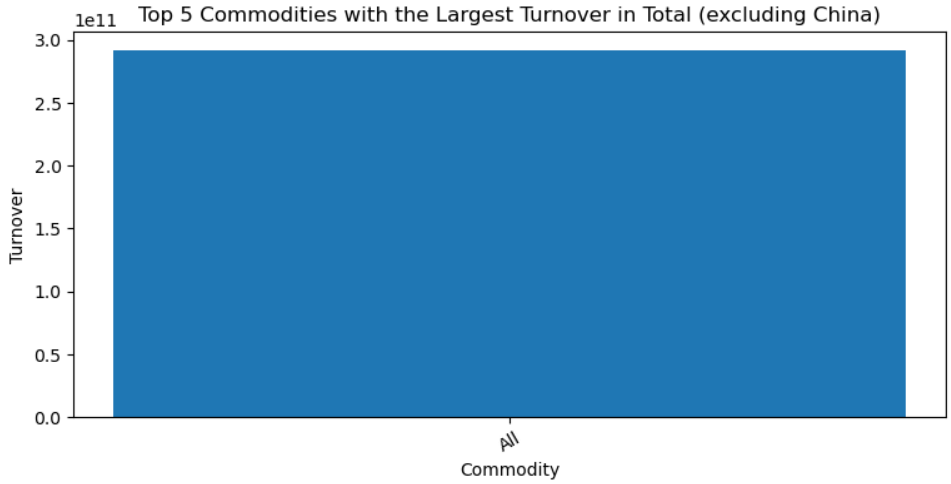

Top 5 Commodities with the Largest Turnover in United States
1e10
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Turnover
All
Meat and edible offal
Commodity


Top 5 Commodities with the Largest Turnover in Japan
1e10
2.0
1.5
1.0
0.5
0.0
Turnover
All
Commodity


Top 5 Commodities with the Largest Turnover in United Kingdom
1e10
2.0
1.5
1.0
0.5
0.0
Turnover
All
Commodity


Top 5 Commodities with the Largest Turnover in European Union (27)
1e10
2.5
2.0
1.5
1.0
0.5
0.0
Turnover
All
Commodity





8). Παρουσίαση της ημέρας με το μεγαλύτερο τζίρο, για κάθε κατηγορία εμπορεύματος

In[22]:

 # filter out rows where the commodity is "All"

df = df[df['Commodity'] != 'All']

Φιλτράρω γραμμές από το πλαίσιο δεδομένων df ,ώστε να αποφύγω την στήλη Commodity έχει την τιμή Όλα.

 # group the data by commodity

groups = df.groupby('Commodity')

Στη συνέχεια, ομαδοποιώ τα δεδομένα κατά εμπόρευμα χρησιμοποιώντας τη μέθοδο groupby, δημιουργώντας μια νέα ομάδα αντικειμένων Pandas GroupBy.

 # create an empty dataframe to hold the results

results = pd.DataFrame(columns=['Commodity', 'Day', 'Turnover'])

Δημιουργώ ένα κενό πλαίσιο δεδομένων με τρεις στήλες: Εμπορεύματα, Ημέρα και Τζίρο.

 # loop through each group and find the day with the highest turnover

for name, group in groups:

 # group the data by date and sum the values for each day

by_date = group.groupby('Date')['Value'].sum()

Περνάω μέσα από κάθε ομάδα , που αντιπροσωπεύει ένα εμπόρευμα, και για κάθε ομάδα, ομαδοποιώ τα δεδομένα κατά ημερομηνία και υπολογίζω το άθροισμα των τιμών για κάθε ημέρα χρησιμοποιώντας τη μέθοδο groupby. Στη συνέχεια ταξινομώ το πλαίσιο δεδομένων που προκύπτει με βάση το άθροισμα των τιμών σε φθίνουσα σειρά και επιλέγω το υψηλότερο άθροισμα, αντιπροσωπεύοντας την ημέρα με τον υψηλότερο τζίρο για αυτό το εμπόρευμα.

 # sort the resulting dataframe by sum of values in descending order

sorted_by_date = by_date.sort_values(ascending=False)

 # select the row with the highest sum

highest_day = sorted_by_date.iloc[0]

 # calculate the percentage of total turnover that occurred on the highest day

total_turnover = group['Value'].sum()

Υπολογίζω τον τζίρο για αυτό το εμπόρευμα αθροίζοντας τη στήλη Αξία για όλες τις γραμμές της ομάδας και υπολογίζω το ποσοστό του συνολικού τζίρου που πραγματοποιήθηκε την υψηλότερη ημέρα διαιρώντας τον τζίρο της υψηλότερης ημέρας με τον συνολικό τζίρο. (Ορίζω ότι η ταξινόμηση πρέπει να γίνεται με φθίνουσα σειρά).

 # add the result to the results dataframe

results = results.append({'Commodity': name, 'Day': sorted_by_date.index[0], 'Turnover': total_turnover}, ignore_index=True)

Προσθέτω τα αποτελέσματα για αυτό το εμπόρευμα στο πλαίσιο δεδομένων χρησιμοποιώντας τη μέθοδο προσάρτησης και το όρισμα ignore_index=True για να διασφαλίσω ότι το ευρετήριο επαναφέρεται.

results

Το results είναι ένα πλαίσιο δεδομένων που περιέχει την κορυφαία ημέρα για κάθε εμπόρευμα, μαζί με τον συνολικό τζίρο για αυτό το εμπόρευμα και το ποσό του συνολικού τζίρου που πραγματοποιήθηκε την κορυφαία ημέρα.

Out[22]:

	Commodity	Day	Turnover
0	Electrical machinery and equip	2020-11-06	51554000000
1	Fish, crustaceans, and molluscs	2017-09-25	15446832000
2	Fruit	2020-05-03	22197000000
3	Logs, wood, and wood articles	2021-07-30	50645402000
4	Meat and edible offal	2017-05-21	78522372000
5	Mechanical machinery and equip	2018-10-02	72603000000
6	Milk powder, butter, and cheese	2019-12-04	157319791000
7	Non-food manufactured goods	2020-06-26	403154000000

In[23]:

plt.figure(figsize=(12,4))

Ορίζω το μέγεθος του plot σε πλάτος 12 ίντσες και ύψος 4 ίντσες.

plt.bar(results['Commodity'], results['Turnover'])

Δημιουργώ μια γραφική παράσταση ράβδων όπου ο άξονας x είναι οι μοναδικές τιμές στη στήλη Commodity του πλαισίου δεδομένων αποτελεσμάτων και ο άξονας y είναι το Στήλη τζίρου του πλαισίου δεδομένων αποτελεσμάτων.

plt.xticks(rotation=90)

Περιστρέφω τις ετικέτες του άξονα x κατά 90 μοίρες για καλύτερη ορατότητα.

plt.ylabel('Total Turnover')

Προσθέτω μια ετικέτα για τον άξονα y.

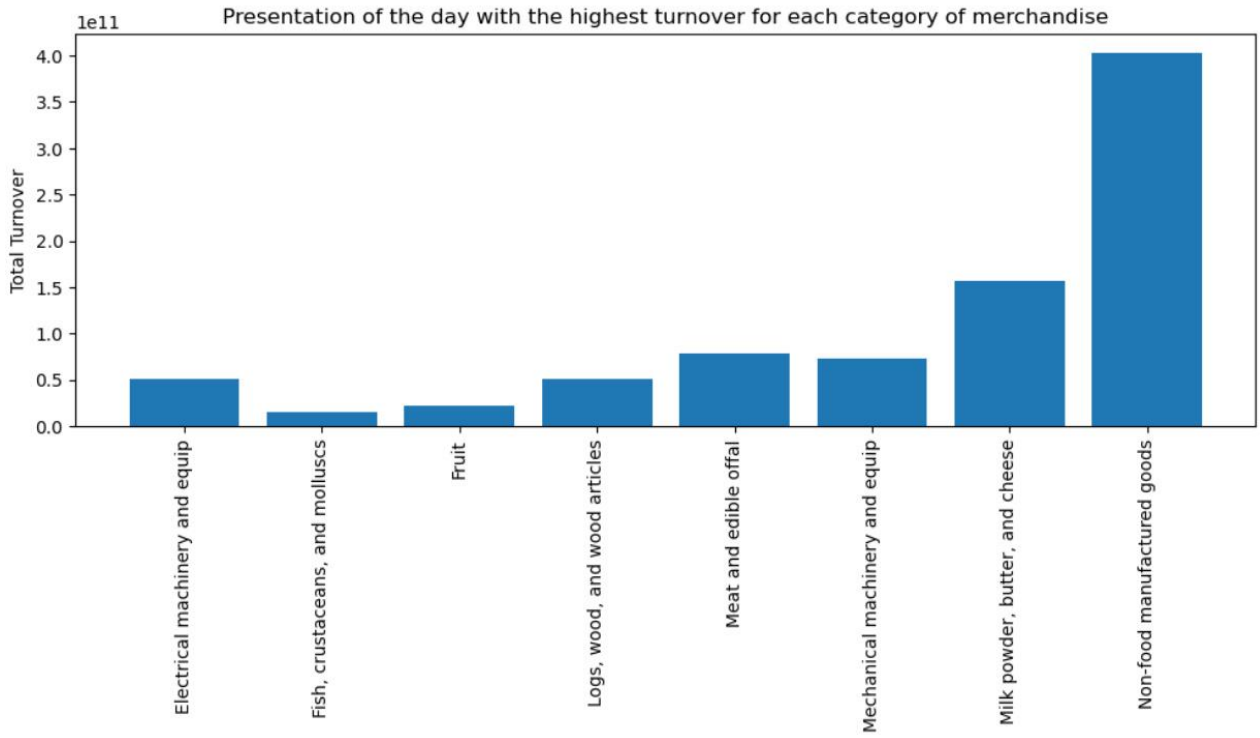
plt.title('Presentation of the day with the highest turnover for each category of merchandise')

Παρουσίαση της ημέρας με τον υψηλότερο τζίρο για κάθε κατηγορία εμπορευμάτων προσθέτει έναν τίτλο στο plot .

plt.show()

Εμφάνιση του γραφήματος.

Δημιουργώ ένα διάγραμμα ράβδων για την εμφάνιση του συνολικού τζίρου για κάθε κατηγορία εμπορευμάτων κατά μήκος του άξονα x. Ο άξονας y αντιπροσωπεύει τον συνολικό τζίρο για κάθε κατηγορία. Το plot έχει τίτλο «Παρουσίαση ημέρας με τον μεγαλύτερο τζίρο για κάθε κατηγορία εμπορευμάτων». Η γραφική παράσταση δημιουργείται χρησιμοποιώντας τη μονάδα pyplot του Matplotlib.



DOWNLOAD.PY

import requests

Στόχος είναι η λήψη ενός αρχείου CSV από μια διεύθυνση URL χρησιμοποιώντας τη βιβλιοθήκη requests στην Python.

url = https://www.stats.govt.nz/assets/Uploads/Effects-of-COVID-19-on-trade/Effects-of-COVID-19-on-trade-At-15-December-2021-provisional/Download-data/effects-of-covid-19-on-trade-at-15-december-2021-provisional.csv

Η μεταβλητή url αποθηκεύει τη διεύθυνση URL όπου βρίσκεται το αρχείο CSV.

filename = "covid_trade_data.csv"

Αποθηκεύω το όνομα του αρχείου που θα αποθηκευτεί μετά τη λήψη.

response = requests.get(url)

Η μέθοδος requests.get() χρησιμοποιείται για τη λήψη των δεδομένων από τη διεύθυνση URL, η οποία επιστρέφει ένα αντικείμενο response.

with open(filename, "wb") as f:

Χρησιμοποιειώ με το όνομα αρχείου και τη λειτουργία "wb" (δυαδική εγγραφή) για να ανοίξει το αρχείο σε δυαδική λειτουργία για εγγραφή.

f.write(response.content)

Λήψη των δυαδικών δεδομένων από το αντικείμενο response. Χρησιμοποιειώ την μέθοδος f.write() για την εγγραφή των δυαδικών δεδομένων στο αρχείο.

print(f"CSV file downloaded and saved as {filename}")

Τέλος, εκτυπώνω ένα μήνυμα στην οθόνη που υποδεικνύει ότι το αρχείο CSV έχει ληφθεί και αποθηκευτεί με το συγκεκριμένο όνομα αρχείου.

MYSQL.PY

import mysql.connector

Εισάγω τη μονάδα σύνδεσης MySQL που επιτρέπει στην Python να αλληλεπιδρά με μια βάση δεδομένων MySQL.

import csv

Επιτρέπω στην Python να διαβάζει και να γράφει αρχεία CSV.

import pandas as pd

Εισάγω τη βιβλιοθήκη Pandas ως pd, η οποία χρησιμοποιείται για εργασία με δομές δεδομένων όπως πλαίσια δεδομένων.

from datetime import datetime

Εισάγω το datetime από την βιβλιοθήκη της Python, η οποία χρησιμοποιείται για εργασία με ημερομηνίες και ώρες.

 # Load dataframe

df = pd.read_csv("covid_trade_data.csv")

Διαβάζω ένα αρχείο CSV που ονομάζεται "covid_trade_data.csv" και το φορτώνω σε ένα πλαίσιο δεδομένων Pandas που ονομάζεται df.

 # convert the date column to the correct format

df['Date'] = df['Date'].apply(lambda x: datetime.strptime(x, '%d/%m/%Y').strftime('%Y-%m-%d'))

Μετατρέπω τη στήλη Ημερομηνία στο πλαίσιο δεδομένων Pandas στη σωστή μορφή ημερομηνίας EEEE-MM-HH χρησιμοποιώντας τις μεθόδους strptime και strftime από τη βιβλιοθήκη datetime.

 # connect to the MySQL database

#

cnx = mysql.connector.connect(user='root', password='bscs3945',

host='localhost',

database='covid_data')

Δημιουργώ μια σύνδεση με την βάση δεδομένων MySQL που ονομάζεται covid_data με όνομα χρήστη roo", κωδικό πρόσβασης "bscs3945" και φιλοξενείται στο localhost.

 # create the necessary tables in the database

cursor = cnx.cursor()

Δημιουργώ έναν κέρσορα που χρησιμοποιείται για την εκτέλεση ερωτημάτων SQL και την ανάκτηση αποτελεσμάτων από τη βάση δεδομένων MySQL.

create_table_query = '''CREATE TABLE IF NOT EXISTS covid90_table (

Direction VARCHAR(255),

Year INT,

Date DATETIME,

```
            Weekday VARCHAR(255),
            Country VARCHAR(255),
            Commodity VARCHAR(255),
            Transport_Mode VARCHAR(255),
            Measure VARCHAR(255),
            Value BIGINT,
            Cumulative BIGINT)'''
cursor.execute(create_table_query)
```

Δημιουργώ έναν νέο πίνακα στη βάση δεδομένων MySQL που ονομάζεται covid90_table με τα καθορισμένα ονόματα στηλών και τους τύπους δεδομένων.

```
# convert the dataframe to a list of tuples
data = [tuple(row) for _, row in df.iterrows()]
```

Μετατρέπω το πλαίσιο δεδομένων Pandas σε μια λίστα πλειάδων, όπου κάθε πλειάδα περιέχει τις τιμές για μία μόνο γραμμή στο πλαίσιο δεδομένων.

```
# insert the data into the corresponding MySQL table
insert_query = '''INSERT INTO covid90_table
(Direction, Year, Date, Weekday, Country, Commodity,
Transport_Mode, Measure, Value, Cumulative)
            VALUES
(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)'''
cursor.executemany(insert_query, data)
cnx.commit()
```

Εκτελώ ένα SQL query για να εισαγάγω τα δεδομένα από τη λίστα των πλειάδων στον πίνακα covid90_table στη βάση δεδομένων MySQL. Τα δεδομένα εισάγονται χρησιμοποιώντας ένα παραμετροποιημένο ερώτημα για την αποτροπή επιθέσεων SQL injection. Οι αλλαγές δεσμεύονται στη βάση δεδομένων χρησιμοποιώντας τη μέθοδο δέσμευσης.

 # export the data from the MySQL table to a .csv file

select_query = "'SELECT * FROM covid90_table'"

Ορίζω μια μεταβλητή συμβολοσειράς select_query στο ερώτημα SQL που επιλέγει όλες τις στήλες και τις σειρές από τον πίνακα covid90_table.

cursor.execute(select_query)

Εκτελώ το ερώτημα SQL χρησιμοποιώντας το κέρσορα που δημιουργήθηκε νωρίτερα.

result = cursor.fetchall()

Ανακτώ όλες τις σειρές που επιστρέφονται από το ερώτημα και τις αποθηκεύω στη μεταβλητή αποτελέσματος.

Εκτελώ ένα ερώτημα SQL για να επιλέξω όλα τα δεδομένα από τον πίνακα covid90_table στη βάση δεδομένων MySQL και να ανακτήσω τα αποτελέσματα σε μια μεταβλητή που ονομάζεται αποτέλεσμα.

with open('covid90_table.csv', 'w', newline='') as csvfile:

Ανοίγω ένα νέο αρχείο με το όνομα covid90_table.csv σε λειτουργία εγγραφής και δημιουργώ ένα νέο αντικείμενο αρχείου csvfile.

csvwriter = csv.writer(csvfile)

Δημιουργώ ένα νέο αντικείμενο csvwriter χρησιμοποιώντας το αντικείμενο αρχείου csvfile.

csvwriter.writerow([i[0] for i in cursor.description])

Γράφω τη γραμμή κεφαλίδας του αρχείου CSV χρησιμοποιώντας τα ονόματα στηλών από το σχήμα του πίνακα. Το χαρακτηριστικό cursor.description περιέχει πληροφορίες σχετικά με τις στήλες στο αποτέλεσμα του ερωτήματος.

csvwriter.writerows(result)

Γράφω τις γραμμές του αποτελέσματος του ερωτήματος στο αρχείο CSV.

Ανοίγω ένα νέο αρχείο CSV που ονομάζεται covid90_table.csv και χρησιμοποιώ το αντικείμενο csvwriter για να γράψω τη στήλη

 # close the database connection

cursor.close()

Κλείνω τον κέρσορα.

cnx.close()

Κλείνω τη σύνδεση της βάσης δεδομένων.

GUI.PY

import pandas as pd

Εκχωρώ το ψευδώνυμο pd στην pandas, ώστε να αναφερόμαστε εύκολα σε αυτή σε ολόκληρο τον κώδικα.

import matplotlib.pyplot as plt

Εκχωρώ το ψευδώνυμο plt στο pyplot της βιβλιοθήκης matplotlib, ώστε να αναφερόμαστε εύκολα σε αυτό σε ολόκληρο τον κώδικα.

import PySimpleGUI as sg

Είναι μια βιβλιοθήκη για GUI στη Python, που παρέχει μια εύχρηστη διεπαφή για γραφικά περιβάλλοντα χρήστη. Εκχωρώ το ψευδώνυμο sg, ώστε να αναφερόμαστε εύκολα σε αυτή σε ολόκληρο τον κώδικα.

 # Load the data

df = pd.read_csv('covid_trade_data.csv')

Φορτώνω δεδομένα από το αρχείο CSV covid_trade_data.csv σε ένα pandas DataFrame - df.

df['Date'] = pd.to_datetime(df['Date'], format='%d/%m/%Y')

Μετατρέπω τη στήλη Date στο DataFrame από μορφή συμβολοσειράς σε μορφή ημερομηνίας, χρησιμοποιώντας τη συνάρτηση pandas 'to_datetime'. Το πρώτο όρισμα στη συνάρτηση είναι η στήλη που θα μετατραπεί και το δεύτερο όρισμα είναι η μορφή των ημερομηνιών ημέρα/μήνας/έτος.

 # Define the functions to create each plot

def plot_monthly_turnover():

Ορίζω μια συνάρτηση plot_monthly_turnover() που δημιουργεί ένα γραμμικό γράφημα δεδομένων για το τζίρο ανά μήνα.

df['Date'] = pd.to_datetime(df['Date'], format='%d/%m/%Y')

Μετατρέπω τη στήλη Date του DataFrame df σε μορφή ημερομηνίας χρησιμοποιώντας το pd.to_datetime(). Η μορφή που χρησιμοποιείται για την ανάλυση των ημερομηνιών καθορίζεται ως '%d/%m/%Y'.

monthly_turnover = df.groupby(pd.Grouper(key='Date', freq='M'))['Value'].sum()

Υπολογίζω τον συνολικό μηνιαίο τζίρο ομαδοποιώντας το DataFrame df ανά μήνα χρησιμοποιώντας το pd.Grouper() με συχνότητα M και, στη συνέχεια, λαμβάνοντας το άθροισμα της στήλης Τιμή για κάθε ομάδα.

plt.plot(monthly_turnover.index, monthly_turnover.values)

Δημιουργώ ένα γραμμικό διάγραμμα των δεδομένων μηνιαίου τζίρου χρησιμοποιώντας το plt.plot(). Ο άξονας x της γραφικής παράστασης αντιστοιχεί στις μηνιαίες ημερομηνίες, στις οποίες προσπελάστηκαν χρησιμοποιώντας το monthly_turnover.index, ενώ ο άξονας y αντιστοιχεί στις συνολικές τιμές μηνιαίου τζίρου, στις οποίες προσπελάστηκαν με τη χρήση **monthly_turnover.values.**

plt.title('Total Turnover per Month')

Ορίζω τον τίτλο της πλοκής ως Συνολικός τζίρος ανά μήνα χρησιμοποιώντας το plt.title().

plt.xlabel('Month')

Ορίζω την ετικέτα του άξονα x ως Month χρησιμοποιώντας το plt.xlabel().

plt.ylabel('Turnover')

Ορίζω την ετικέτα του άξονα y ως τζίρος χρησιμοποιώντας plt.ylabel().

plt.show()

Εμφανίζω το plot χρησιμοποιώντας plt.show().

def plot_country_turnover():

Ορίζω την συνάρτηση plot_country_turnover() που δημιουργεί ένα γράφημα ράβδων που δείχνει τον τζίρο για κάθε χώρα στο σύνολο δεδομένων.

country_turnover = df.groupby('Country')['Value'].sum()

Ομαδοποιώ το DataFrame df ανά χώρα και στη συνέχεια υπολογίζω το άθροισμα της στήλης Value για κάθε ομάδα.

plt.figure(figsize=(12,5))

Εκχωρώ το μέγεθος σε 12 επί 5 ίντσες.

plt.bar(country_turnover.index, country_turnover.values)

Δημιουργώ ένα γράφημα ράβδων χρησιμοποιώντας το index of country_turnover (δηλαδή τα ονόματα των χωρών) ως ετικέτες του άξονα x και τις τιμές του country_turnover (δηλαδή τον συνολικό τζίρο) ως τα ύψη των ράβδων.

plt.title('Total Turnover per Country')

Ορίζω τον τίτλο της πλοκής ως Συνολικός τζίρος ανά χώρα χρησιμοποιώντας το plt.title().

plt.xlabel('Country')

Ορίζω την ετικέτα του άξονα x ως Country χρησιμοποιώντας το plt.xlabel().

plt.ylabel('Turnover')

Ορίζω την ετικέτα του άξονα y ως τζίρος χρησιμοποιώντας plt.ylabel().

plt.xticks(rotation=30)

Περιστρέφω τις ετικέτες του άξονα x κατά 30 μοίρες για να τις διευκολύνω στην ανάγνωση.

plt.show()

Εμφανίζω το plot χρησιμοποιώντας plt.show().

def plot_transport_turnover():

Ορίζω την συνάρτηση plot_transport_turnover και δημιουργώ ένα γράφημα ράβδων που δείχνει τον τζίρο ανά μέσο μεταφοράς για τα δεδομένα που είναι αποθηκευμένα στο πλαίσιο δεδομένων df.

transport_turnover = df.groupby('Transport_Mode')['Value'].sum()

Ομαδοποιώ πρώτα τα δεδομένα σε df με τη στήλη Transport_Mode χρησιμοποιώντας τη μέθοδο groupby() και στη

συνέχεια υπολογίζω τον συνολικό τζίρο για κάθε μέσο μεταφοράς χρησιμοποιώντας τη μέθοδο sum() στη στήλη Value.

plt.bar(transport_turnover.index, transport_turnover.values)

Δημιουργώ ένα γράφημα ράβδων χρησιμοποιώντας τη συνάρτηση plt.bar() από τη βιβλιοθήκη matplotlib.pyplot, με τον άξονα x να δείχνει τους διαφορετικά μέσα μεταφοράς και τον άξονα y να δείχνει τις αντίστοιχες τιμές των εσόδων.

plt.title('Overall Turnover per Means of Transport')

Ορίζω τον τίτλο της πλοκής ως Συνολικός τζίρος ανά μέσο μεταφοράς χρησιμοποιώντας το plt.title().

plt.xlabel('Transport Mode')

Ορίζω την ετικέτα του άξονα x ως Transport Mode χρησιμοποιώντας το plt.xlabel().

plt.ylabel('Turnover')

Ορίζω την ετικέτα του άξονα y ως τζίρος χρησιμοποιώντας plt.ylabel().

plt.show()

Εμφανίζω το plot χρησιμοποιώντας plt.show().

def plot_weekday_turnover():

Ορίζω την συνάρτηση plot_weekday_turnover(), η οποία δημιουργεί μια γραφική παράσταση ράβδων που δείχνει τον συνολικό τζίρο για κάθε ημέρα της εβδομάδας.

weekday_turnover = df.groupby('Weekday')['Value'].sum()

Υπολογίζω πρώτα τον συνολικό τζίρο για κάθε ημέρα της εβδομάδας ομαδοποιώντας τα δεδομένα στο df DataFrame με τη στήλη «Weekday» και αθροίζοντας τη στήλη «Value»

χρησιμοποιώντας τις μεθόδους .groupby() και .sum(). Το αποτέλεσμα αποθηκεύεται στη μεταβλητή weekday_turnover.

weekday_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']

Ορίζω μια λίστα που ονομάζεται weekday_order που καθορίζει τη σειρά με την οποία θα πρέπει να εμφανίζονται οι ημέρες της εβδομάδας στην γραφική παράσταση.

weekday_turnover = weekday_turnover.reindex(weekday_order)

Η σειρά weekday_turnover αναπροσαρμόζεται με βάση τη λίστα weekday_order χρησιμοποιώντας τη μέθοδο .reindex(), η οποία διασφαλίζει ότι οι ημέρες της εβδομάδας εμφανίζονται με την καθορισμένη σειρά στην γραφική παράσταση.

plt.bar(weekday_turnover.index, weekday_turnover.values)

Δημιουργώ ένα διάγραμμα ράβδων χρησιμοποιώντας το plt.bar(), όπου ο άξονας x δείχνει τις ημέρες της εβδομάδας και ο άξονας y δείχνει τον τζίρο.

plt.title('Total Turnover per Day of the Week')

Ορίζω τον τίτλο της πλοκής ως Συνολικός τζίρος ανά ημέρα της εβδομάδας χρησιμοποιώντας το plt.title().

plt.xlabel('Day of the Week')

Ορίζω την ετικέτα του άξονα x ως Day of the Week χρησιμοποιώντας το plt.xlabel().

plt.ylabel('Turnover')

Ορίζω την ετικέτα του άξονα y ως τζίρος χρησιμοποιώντας plt.ylabel().

plt.xticks(rotation=30)

Περιστρέφω τις ετικέτες του άξονα x κατά 30 μοίρες για να τις διευκολύνω στην ανάγνωση.

plt.show()

Εμφανίζω το plot χρησιμοποιώντας plt.show().

def plot_commodity_turnover_category_of_good():

Δημιουργώ ένα οριζόντιο διάγραμμα ράβδων που δείχνει τον συνολικό τζίρο ανά κατηγορία αγαθών στο σύνολο δεδομένων.

commodity_turnover = df.groupby('Commodity')['Value'].sum()

Υπολογίζω τον συνολικό τζίρο ανά κατηγορία αγαθών ομαδοποιώντας το πλαίσιο δεδομένων με βάση τη στήλη Commodity και αθροίζοντας τη στήλη Αξία για κάθε ομάδα. Αυτό γίνεται χρησιμοποιώντας τις μεθόδους groupby() και sum() ενός pandas DataFrame.

plt.barh(commodity_turnover.index, commodity_turnover.values)

Δημιουργώ μια οριζόντια γραφική γραμμή χρησιμοποιώντας plt.barh(). Αυτή η μέθοδος παίρνει τα ονόματα των κατηγοριών ως τιμές του άξονα y και τις τιμές τζίρου ως τιμές του άξονα x. Η παράμετρος δείκτης ορίζεται στα ονόματα των κατηγοριών και η παράμετρος τιμών ορίζεται στις τιμές κύκλου εργασιών. Αυτό θα δημιουργήσει οριζόντιες ράβδους για κάθε κατηγορία αγαθών, με το μήκος κάθε ράβδου ανάλογο με τον συνολικό κύκλο εργασιών για αυτήν την κατηγορία.

plt.title('Total Turnover per Category of Goods')

Ορίζω τον τίτλο της πλοκής ως Συνολικός τζίρος ανά κατηγορία αγαθών χρησιμοποιώντας το plt.title().

plt.ylabel('Category of Goods')

Ορίζω την ετικέτα του άξονα y ως Category of Goods χρησιμοποιώντας plt.ylabel().

plt.xlabel('Turnover')

Ορίζω την ετικέτα του άξονα x ως Turnover χρησιμοποιώντας το plt.xlabel().

plt.show()

Εμφανίζω το plot χρησιμοποιώντας plt.show().

def plot_monthly_top_5_turnover():

Δημιουργώ ένα γράφημα ράβδων που δείχνει τους κορυφαίους 5 μήνες με τον υψηλότερο τζίρο.

monthly_turnover = df.groupby(pd.Grouper(key='Date', freq='M'))['Value'].sum()

Υπολογίζω τον μηνιαίο τζίρο ομαδοποιώντας το DataFrame df ανά μήνα χρησιμοποιώντας τη συνάρτηση pd.Grouper() και τη μέθοδο sum() για την άθροιση του κύκλου εργασιών για κάθε μήνα.

top_months = monthly_turnover.sort_values(ascending=False)[:5]

Επιλέγω τους κορυφαίους 5 μήνες με τον υψηλότερο τζίρο χρησιμοποιώντας τη μέθοδο sort_values() για να ταξινομήσω τον μηνιαίο τζίρο με φθίνουσα σειρά και τον τελεστή slice [:5] για να επιλέξω τις πρώτες 5 γραμμές του DataFrame που προκύπτει.

plt.bar(top_months.index.strftime('%B %Y'), top_months.values)

Δημιουργώ μια γραφική παράσταση οριζόντιας ράβδου χρησιμοποιώντας τη μέθοδο bar() της μονάδας pyplot του Matplotlib. Ο άξονας x δείχνει τον τζίρο για κάθε έναν από τους κορυφαίους 5 μήνες και ο άξονας y δείχνει το όνομα του μήνα.

plt.title('5 Months with the Highest Turnover')

Ορίζω τον τίτλο της πλοκής ως 5 Μήνες με τον υψηλότερο τζίρο χρησιμοποιώντας το plt.title().

plt.xlabel('Month')

Ορίζω την ετικέτα του άξονα x ως Month χρησιμοποιώντας το plt.xlabel().

plt.ylabel('Turnover')

Ορίζω την ετικέτα του άξονα y ως 'Turnover' χρησιμοποιώντας το ply.xlabel().

plt.xticks(rotation=30)

Περιστρέφω τις ετικέτες του άξονα x κατά 30 μοίρες για να τις διευκολύνω στην ανάγνωση και να αποφύγω την επικάλυψη ετικετών.

plt.show()

Εμφανίζω το plot χρησιμοποιώντας plt.show().

def plot_top_5_country_commodity_turnover():

Ορίζω την plot_top_5_country_commodity_turnover που απεικονίζει τα κορυφαία 5 εμπορεύματα με τον τζίρο για κάθε χώρα στο σύνολο δεδομένων.

country_commodity_turnover = df.groupby(['Country', 'Commodity'])['Value'].sum()

Ομαδοποιώ το πλαίσιο δεδομένων κατά Χώρα και Εμπόρευμα και συγκεντρώνοντας το άθροισμα της Αξίας για κάθε ομάδα χρησιμοποιώντας τη μέθοδο groupby. Αυτό δημιουργεί μια νέα σειρά με ένα ιεραρχικό ευρετήριο που αποτελείται από Χώρα και Εμπόρευμα. Στη συνέχεια, αυτή η σειρά χρησιμοποιείται για τον προσδιορισμό των 5 κορυφαίων εμπορευμάτων με τον μεγαλύτερο τζίρο για κάθε χώρα.

 # Loop over the unique countries in the data

for country in df['Country'].unique():

Εκτελώ έναν βρόγχο πάνω από τις μοναδικές χώρες στο πλαίσιο δεδομένων χρησιμοποιώντας τη μέθοδο unique και δημιουργώ ένα νέο σχήμα για κάθε χώρα.

 # Get the top 5 commodities for the current country

top_commodities = country_commodity_turnover.loc[country].nlargest(5)

Μέσα σε κάθε βρόχο, τα 5 κορυφαία εμπορεύματα με τον μεγαλύτερο κύκλο εργασιών για την τρέχουσα χώρα εξάγονται χρησιμοποιώντας τη μέθοδο loc με δείκτη την τρέχουσα χώρα.

 # Plot the results for the current country

plt.figure(figsize=(9,4))

plt.bar(top_commodities.index, top_commodities.values)

plt.title(f'Top 5 Commodities with the Largest Turnover in {country}')

plt.xlabel('Commodity')

plt.ylabel('Turnover')

plt.xticks(rotation=30)

plt.show()

Δημιουργώ ένα οριζόντιο διάγραμμα ράβδων για κάθε χώρα με τα κορυφαία εμπορεύματα στον άξονα y και τις αντίστοιχες τιμές τζίρου στον άξονα x. Το γράφημα επισημαίνεται με το όνομα της χώρας και ο τίτλος υποδηλώνει ότι δείχνει τα κορυφαία 5 εμπορεύματα με τον μεγαλύτερο τζίρο σε αυτήν τη χώρα. Οι ετικέτες του άξονα x περιστρέφονται για καλύτερη ορατότητα χρησιμοποιώντας τη λειτουργία xticks.

df = df[df['Commodity'] != 'All']

Αρχικά φιλτράρω τις σειρές όπου η στήλη Εμπορεύματα έχει την τιμή Όλα και αποθηκεύω το DataFrame που προκύπτει πίσω στο df.

groups = df.groupby('Commodity')

Στη συνέχεια, ομαδοποιώ τις σειρές του DataFrame με τις τιμές στη στήλη Commodity και αποθηκεύω τις ομάδες που προκύπτουν στο αντικείμενο ομάδων.

def highest_turnover_for_each_category_of_merchandise():

Η high_turnover_for_each_category_of_merchandise() περνά μέσα από κάθε ομάδα σε ομάδες και εκτελεί τις ακόλουθες ενέργειες:

results = pd.DataFrame(columns=['Commodity', 'Day', 'Turnover'])

for name, group in groups:

by_date = group.groupby('Date')['Value'].sum()

Υπολογίζω τον συνολικό τζίρο για κάθε ημερομηνία εντός της ομάδας χρησιμοποιώντας τη μέθοδο groupby() και αποθηκεύω τη σειρά που προκύπτει στο αντικείμενο by_date.

sorted_by_date = by_date.sort_values(ascending=False)

Ταξινομώ τη σειρά by_date σε φθίνουσα σειρά κατά τιμές χρησιμοποιώντας τη μέθοδο sort_values() και αποθηκεύω τη σειρά που προκύπτει στο αντικείμενο sorted_by_date.

highest_day = sorted_by_date.iloc[0]

Επιλέγω την ημερομηνία με τον υψηλότερο τζίρο χρησιμοποιώντας τη μέθοδο .iloc[0] στη σειρά sorted_by_date και αποθηκεύω την τιμή που προκύπτει στη μεταβλητή highest_day.

total_turnover = group['Value'].sum()

Υπολογίζω τον συνολικό τζίρο για την τρέχουσα ομάδα αθροίζοντας τις τιμές στη στήλη Τιμή του αντικειμένου ομάδας.

new_row = pd.DataFrame({'Commodity': [name], 'Day': [sorted_by_date.index[0]], 'Turnover': [total_turnover]})

Δημιουργώ ένα νέο αντικείμενο DataFrame με τις στήλες Commodity, Day και Turnover και τις τιμές από την τρέχουσα ομάδα χρησιμοποιώντας τη μέθοδο pd.DataFrame().

results = pd.concat([results, new_row], ignore_index=True)

Συνδέω το νέο αντικείμενο DataFrame με το αντικείμενο DataFrame αποτελεσμάτων χρησιμοποιώντας τη μέθοδο pd.concat().Το προκύπτον αντικείμενο DataFrame θα έχει μια γραμμή για κάθε τιμή Εμπορεύματος, με την υψηλότερη ημέρα κύκλου εργασιών και τον συνολικό τζίρο για αυτό το εμπόρευμα.

plt.figure(figsize=(12,4))

Ορίζω το μέγεθος του σχήματος της γραφικής παράστασης σε 12 ίντσες (πλάτος) επί 4 ίντσες (ύψος).

plt.bar(results['Commodity'], results['Turnover'])

Τα δεδομένα που θα απεικονιστούν λαμβάνονται από τα αποτελέσματα DataFrame που δημιουργήθηκε στο προηγούμενο μπλοκ κώδικα, όπου κάθε σειρά αντιπροσωπεύει μια κατηγορία εμπορευμάτων, την ημέρα με τον υψηλότερο τζίρο και τον συνολικό τζίρο για αυτήν την κατηγορία.

Χρησιμοποιώ τη συνάρτηση plt.bar() για τη δημιουργία ενός ραβδωτού γραφήματος με την κατηγορία των εμπορευμάτων στον άξονα x και τον συνολικό τζίρο στον άξονα y.

plt.xticks(rotation=90)

Περιστροφή των ετικετών του άξονα x κατά 90 μοίρες για την αποφυγή επικάλυψης.

plt.ylabel('Total Turnover')

plt.title('Presentation of the day with the highest turnover for each category of merchandise')

plt.show()

Εμφάνιση της γραφικής παράστασης

 # Define the GUI layout

layout = [

Αυτό ορίζει μια λίστα που ονομάζεται διάταξη που θα κρατά τη διάταξη του GUI.

[sg.Text('Select a plot:', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο Text στο GUI που εμφανίζει το μήνυμα "Select a plot:". Ορίζει το μέγεθος του στοιχείου σε πλάτος 20 χαρακτήρων και ύψος 1 χαρακτήρα.

[sg.Button('Monthly Turnover', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο Button στο GUI με την ένδειξη "Monthly Turnover".

[sg.Button('Country Turnover', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο Button στο GUI με την ένδειξη "Country Turnover".

[sg.Button('Transport Turnover', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο Button στο GUI με την ένδειξη "Transport Turnover".

[sg.Button('Weekday Turnover', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο Button στο GUI με την ένδειξη "Weekday Turnover".

[sg.Button('Total Turnover per Category of Goods', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο κουμπιού στο γραφικό περιβάλλον εργασίας με την ένδειξη "Συνολικός τζίρος ανά κατηγορία αγαθών".

[sg.Button('5 Months with the Highest Turnover', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο κουμπιού στο γραφικό περιβάλλον εργασίας με την ένδειξη "5 μήνες με τον υψηλότερο τζίρο".

[sg.Button('Top 5 Commodities by Country', size=(20, 1))],

Αυτή η γραμμή προσθέτει ένα στοιχείο Button στο GUI με την ένδειξη "Top 5 Commodities ανά χώρα".

[sg.Button('Presentation of the day with the highest turnover for each category of merchandise', size=(20, 1))]

Αυτή η γραμμή προσθέτει ένα στοιχείο Button στο GUI με την ένδειξη "Παρουσίαση της ημέρας με τον υψηλότερο τζίρο για κάθε κατηγορία εμπορευμάτων».

]

Αυτός ο κώδικας ορίζει τη διάταξη για μια γραφική διεπαφή χρήστη (GUI) χρησιμοποιώντας τη βιβλιοθήκη PySimpleGUI. Η διάταξη αποτελείται από πολλά κουμπιά, το καθένα αντιπροσωπεύει μια διαφορετική γραφική παράσταση που μπορεί να δημιουργηθεί από το πρόγραμμα. Τα κουμπιά εμφανίζονται κάθετα στην οθόνη, με μια ετικέτα κειμένου δίπλα σε κάθε κουμπί που υποδεικνύει την γραφική παράσταση που θα δημιουργήσει. Όταν κάνετε κλικ σε ένα κουμπί, θα δημιουργηθεί η αντίστοιχη γραφική παράσταση και θα εμφανιστεί στην οθόνη.

```
# Create the GUI window
window = sg.Window('Data Visualizations', layout)
```

Αυτή η γραμμή κώδικα δημιουργεί ένα παράθυρο για την εφαρμογή GUI (γραφική διεπαφή χρήστη). Ο τίτλος του παραθύρου είναι "Οπτικοποιήσεις δεδομένων" και η διάταξη παραθύρου ορίζεται από τη μεταβλητή διάταξης, η οποία είχε οριστεί προηγουμένως. Το παράθυρο που δημιουργήθηκε αποθηκεύεται στη μεταβλητή παραθύρου.

```
# Loop through events in the GUI window
while True:
    event, values = window.read()
    if event == sg.WIN_CLOSED:
        break
    elif event == 'Monthly Turnover':
        plot_monthly_turnover()
        break
    elif event == 'Country Turnover':
        plot_country_turnover()
        break
    elif event == 'Transport Turnover':
        plot_transport_turnover()
        break
    elif event == 'Weekday Turnover':
        plot_weekday_turnover()
        break
    elif event == 'Total Turnover per Category of Goods':
```

plot_commodity_turnover_category_of_good()

break

elif event == '5 Months with the Highest Turnover':

plot_monthly_top_5_turnover()

break

elif event == 'Top 5 Commodities by Country':

plot_top_5_country_commodity_turnover()

break

elif event == 'Presentation of the day with the highest turnover for each category of merchandise':

highest_turnover_for_each_category_of_merchandise()

break

Αυτός ο κώδικας δημιουργεί έναν βρόχο που ακούει συνεχώς συμβάντα (π.χ. κλικ κουμπιών) στο παράθυρο GUI. Όταν συμβαίνει ένα συμβάν, καλείται η αντίστοιχη συνάρτηση γραφικής παράστασης και ο βρόχος διακόπτεται με την εντολή break.

Για παράδειγμα, εάν κάνετε κλικ στο κουμπί 'Monthly Turnover', καλείται η συνάρτηση plot_monthly_turnover() και ο βρόχος διακόπτεται έτσι ώστε το παράθυρο να κλείσει και να εμφανιστεί η γραφική παράσταση. Ομοίως, για κάθε κουμπί, υπάρχει μια σχετική συνάρτηση γραφικής παράστασης που θα εκτελεστεί όταν κάνετε κλικ στο κουμπί.

 # Close the GUI window

window.close()

Αυτή η γραμμή κώδικα απλώς κλείνει το παράθυρο του GUI μόλις τελειώσει ο βρόχος while. Αυτό διασφαλίζει ότι το

παράθυρο είναι σωστά κλειστό και ότι τυχόν πόροι που σχετίζονται με αυτό απελευθερώνονται.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Η ανάλυση των εμπορικών δεδομένων είναι απαραίτητη για την κατανόηση της απόδοσης του εμπορικού κλάδου σε μια συγκεκριμένη περίοδο. Σε αυτήν την αναφορά, δείξαμε πώς αναλύσαμε ένα σύνολο δεδομένων συναλλαγών και αντλήσαμε σημαντικές πληροφορίες από αυτό. Οι γνώσεις που αποκτήθηκαν μπορούν να χρησιμοποιηθούν από τους ενδιαφερόμενους στον κλάδο του εμπορίου για τη λήψη τεκμηριωμένων αποφάσεων. Είναι σημαντικό να σημειωθεί ότι η ανάλυση μπορεί να προσαρμοστεί ανάλογα με βάση τις ιδιαιτερότητες του συνόλου δεδομένων και την περίπτωση χρήσης του.

Βιβλιογραφία	Βιβλιογραφία
Google's Python Class | Python Education | Google Developers http://unboundprometheus.com/ Python Tutorial (w3schools.com) Welcome | Data Science at the Command Line, 2e (jeroenjanssens.com) https://learn.microsoft.com/en-us/training/paths/data-analytics-microsoft/	Ένα από τα πιο πρόσφατα επιτεύγματά μου ήταν η επιλογή μου για το πρόγραμμα υποτροφιών UNBOUND PROMETHEUS DATA SCIENCE. Συμμετείχα επίσης σε διάφορα εκπαιδευτικά προγράμματα της Microsoft, Google και άλλων φορέων , ώστε να ολοκληρώσω την εκπαίδευση σχετικά με τις απαιτήσεις του project.