

**Pachete Software - Proiect Python:**

[**1) Tratarea calității datelor 2**](#_dh8xin4k61kz)

[**2) Utilizarea listelor 4**](#_bu5aq0pobyhb)

[a) Să se se extragă toate cărțile din categoria fantasy și să fie convertite într-o listă, apoi să fie copiate în cadrul altei liste: 4](#_tugw3elwyt7t)

[b) Să se itereze prin listă și să se verifice dacă există cărți publicate de edituri multiple (nume care se repetă) 5](#_ao6x6va4cgk)

[c) Firma doreșste ca în viitor să comercialize alte 3 cărți, acestea se regăsesc într-o listă diferită și trebuiesc adăugate în lista folosită anterior: 5](#_7tlqqhbur058)

[d) Firma se răzgândește cu privire la comercializarea acelor cărți și le vrea eliminate. 6](#_yplrmf1sczfn)

[e) Se dorește ca cărțile să fie sortate descrescător în funcție de dimensiunea titlului. 6](#_49423mr639o4)

[**3) Utilizarea Tuplurilor 6**](#_yw6r0w9v33pe)

[a) Să se creeze un tuplu care să conțină titlul și autorul cărților din categoria clasici iar apoi să se verifice câte cărți de la fiecare autor sunt disponibile în librărie. 6](#_1bj4r7j2jw1k)

[b) Firma dorește să comercializeze două cărți noi, să se aduage în tuplu. 7](#_x3gd7t6cgiqn)

[**4) Utilizarea Seturilor 8**](#_cxciyoa7xmuo)

[a) Crearea și afisațrea seturilor: 8](#_qpo3fa5nsk12)

[b) Diferența 8](#_fqdxktxl3lv4)

[c) Reuniune 9](#_m83x6nv343l7)

[d) Intersecție 9](#_1r8jnbxceqe5)

[e) Diferență simetrică (cărțile care nu sunt comune celor doua seturi) 9](#_mr20xk6no612)

[**5) Utilizarea dicționarelor 10**](#_z9m7vykd8py)

[a) Să se creeze o listă de dicționare pe baza coloanelor indexurilor coloanelor: 1, 2, 3, 5, 6, 7, 8 10](#_5a9r02irfbnq)

[b) Să se creeze o funcție care calculează profitul generat de fiecare carte din lista de dicționare și să se afișeze acesta. 10](#_xy0x5p257eoc)

[c) Folosind metodele specifice dicționarelor să se afișeze cărțile scrise de fiecare autor. 11](#_ol2l73sazf7y)

[d) Să se elimine din lista de dicționare toate cărțile care au înregistrat un profit negativ: 11](#_af4c2adbcqj7)

[**6) Folosirea merge-ului, grupării și vizualizarea datelor. 12**](#_anxg1i9beuep)

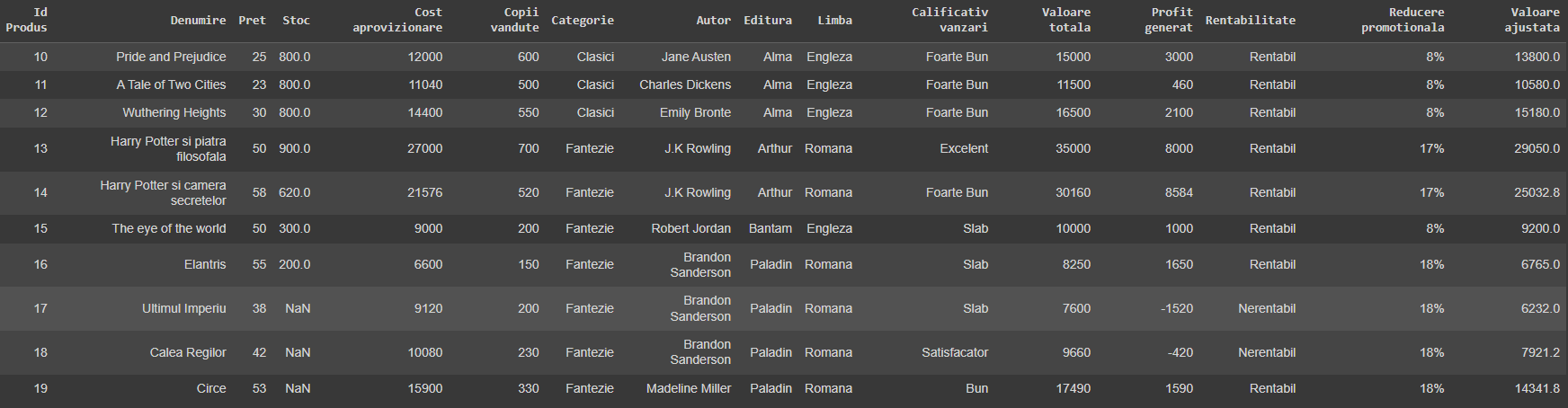
# 1) Tratarea calității datelor

**Descrierea problemei:**

Să se importe fișierul csv care conține date despre vânzările produselor firmei și apoi să se trateze valorile lipsă, să se elimine toate cărțile publicate de editura ‘Patrimoniu’ și să se șteargă coloanele ‘Reducere Promoțională’ și ‘Valoare Ajustată’.

**Informații necesare pentru rezolvare:**

Forma inițială a dataframe-uluil în urma importului este următoarea:



**Figura 1: Dataframe vânzări produse**

**Metoda de rezolvare folosită:**

Înlocuirea valorilor cantitative lipsă cu media pe coloană, ștergerea de înregistrări și coloane.

**Rezolvarea cu ajutorul produsului software**

Citirea fișierului csv a fost realizată prin următoarea linie de cod:

df = pd.read\_csv('date\_proiect\_ps1.csv', encoding='ISO-8859-2')

Apoi am afișat toate valorile înregistrările care prezintă valori nule în coloana Stoc:

print(df[df['Stoc'].isnull()])

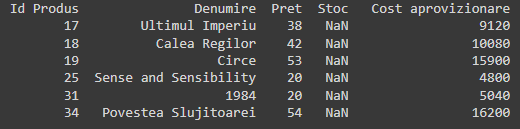


Figura 2: Carțile care prezinta valori lipsă

Tratarea valorilor lipsită a fost realizată prin următoarele linii de cod:

before = df[df['Stoc'].isnull()]

df['Stoc'].fillna(df['Stoc'].mean(), inplace=True)

after = df[df['Stoc'].isnull()]

final = before[~before.index.isin(after.index)]

if not final.empty:

print(f"Valoarea stocului produselor cu ID-ul {', '.join(final['Id Produs'].astype(str).tolist())} a fost schimbata.")

else:

print("Nici o înregistrare nu a fost modificată.")

Cu următorul output:



Schimbarea poate fi văzută rulând următoarea linie de cod:

print(df.loc[df['Id Produs'].isin([17, 18, 19, 25, 31, 34])])

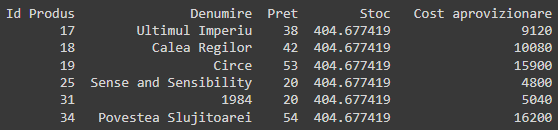


Figura 3: Înlocuirea valorilor lipsă cu media

Afișarea și eliminarea cărților de la editura Patrimoniu a fost realizată prin următorul cod:

patrimoniu = df[df['Editura'] == 'Patrimoniu']

print(patrimoniu)

df.drop(patrimoniu.index, inplace=True)

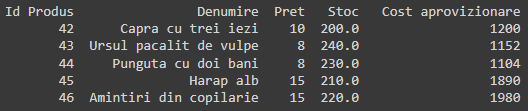


Figura 4: Cărțile editurii Patrimoniu

Eliminarea coloanelor precizate a fost realizată prin următoarea linie de cod:

df.drop(columns=['Reducere promotionala', 'Valoare ajustata'], inplace=True)



Figura 5: Dataframe-ul în urmă curățării datelor

# 2) Utilizarea listelor

## a) Să se se extragă toate cărțile din categoria fantasy și să fie convertite într-o listă, apoi să fie copiate în cadrul altei liste:

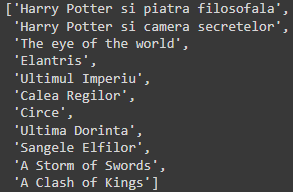
**Cod:**

listaFantezie = df.loc[df['Categorie'] == 'Fantezie', 'Denumire'].tolist()

listaModificari = listaFantezie.copy()

listaModificari

**Output:**



## b) Să se itereze prin listă și să se verifice dacă există cărți publicate de edituri multiple (nume care se repetă)

**Cod:**

counter = 0

for carte in listaModificari:

inventarCarte = listaModificari.count(carte)

if inventarCarte > 1:

print(f'Cartea {carte}! se gaseste in stoc de la {inventarCarte} edituri diferite')

counter = counter + 1

if counter == 0:

print('In lista nu exista carti publicate de edituri diferite')

**Output:**



## c) Firma doreșste ca în viitor să comercialize alte 3 cărți, acestea se regăsesc într-o listă diferită și trebuiesc adăugate în lista folosită anterior:

listaCartiDorite = ['The gardens of the moon','The lies of Locke Lamora','The name of the wind']

#modalitatea 1

for carte in listaCartiDorite:

listaModificari.append(carte)

print(listaModificari)

#stergere carti adauage

for carte in listaCartiDorite:

listaModificari.remove(carte)

#modalitatea 2

listaModificari.extend(listaCartiDorite)

print(listaModificari)

**Output:**

['Harry Potter si piatra filosofala', 'Harry Potter si camera secretelor', 'The eye of the world', 'Elantris', 'Ultimul Imperiu', 'Calea Regilor', 'Circe', 'Ultima Dorinta', 'Sangele Elfilor', 'A Storm of Swords', 'A Clash of Kings', 'The gardens of the moon', 'The lies of Locke Lamora', 'The name of the wind']

## d) Firma se răzgândește cu privire la comercializarea acelor cărți și le vrea eliminate.

**Cod:**

listaModificari.index('The gardens of the moon')

while listaModificari and listaModificari.index(listaModificari[-1]) > 10:

last\_element = listaModificari.pop()

print(listaModificari)

**Output:**

['Harry Potter si piatra filosofala', 'Harry Potter si camera secretelor', 'The eye of the world', 'Elantris', 'Ultimul Imperiu', 'Calea Regilor', 'Circe', 'Ultima Dorinta', 'Sangele Elfilor', 'A Storm of Swords', 'A Clash of Kings']

## e) Se dorește ca cărțile să fie sortate descrescător în funcție de dimensiunea titlului.

**Cod:**

lungime = lambda a: len(a)

print(listaModificari)

listaModificari.sort(key=lungime, reverse=True)

print(listaModificari)

# 3) Utilizarea Tuplurilor

## a) Să se creeze un tuplu care să conțină titlul și autorul cărților din categoria clasici iar apoi să se verifice câte cărți de la fiecare autor sunt disponibile în librărie.

**Cod:**

tupluAutori = tuple(df.loc[df['Categorie'] == 'Clasici', ['Denumire', 'Autor']].apply(tuple, axis=1))

print(tupluAutori)

counter = {}

for denumire, autor in tupluAutori:

if autor in counter:

counter[autor] += 1

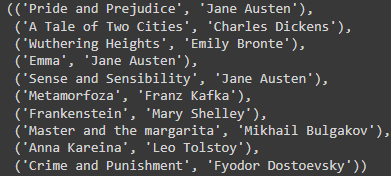
else:

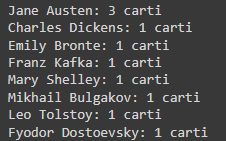
counter[autor] = 1

for autor, count in counter.items():

print(f"{autor}: {count} carti")

**Output:**





## b) Firma dorește să comercializeze două cărți noi, să se aduage în tuplu.

**Cod:**

cartiNoi = (

('The Picture of Dorian Gray', 'Oscar Wilde'),

('Mrs Dalloway', 'Virginia Woolf')

)

tupluAutori += cartiNoi

print(tupluAutori)

**Output:**

(('Sense and Sensibility', 'Jane Austen'), ('A Tale of Two Cities', 'Charles Dickens'), ('Emma', 'Jane Austen'), ('Master and the margarita', 'Mikhail Bulgakov'), ('Metamorfoza ', 'Franz Kafka'), ('Frankestein', 'Mary Shelley'), ('Wuthering Heights', 'Emily Bronte'), ('Anna Kareina', 'Leo Tolstoy'), ('Crime and Punishment', 'Fyodor Dostoevsky'), ('Pride and Prejudice', 'Jane Austen'), ('The Picture of Dorian Gray', 'Oscar Wilde'), ('Mrs Dalloway', 'Virginia Woolf'))

# 4) Utilizarea Seturilor

**Descrierea problemei:**

Firma dorește să se realizeze două seturi, primul să conține primele 10 cărți în funcție de numărul vânzărilor iar cel de-al doilea primele 10 cărți în funcție de profitul generat firmei.

Să se folosească metode specifice seturilor pentru a descoperi diferențele și similaritățile dintre cele două seturi

**Rezolvarea cu ajutorul produsului software:**

## a) Crearea și afisațrea seturilor:

Cod:

top\_10\_vanzari = set(df.nlargest(10, 'Copii vandute')['Denumire'])

top\_10\_profit = set(df.nlargest(10, 'Profit generat')['Denumire'])

print(top\_10\_vanzari)

print(top\_10\_profit)

**Output:**

Cele mai bine vândute:

{'A Clash of Kings', 'Jocurile Foamei', 'A Tale of Two Cities', 'A Storm of Swords', 'Harry Potter si piatra filosofala', 'Pride and Prejudice', 'Crime and Punishment', '1984', 'Wuthering Heights', 'Harry Potter si camera secretelor'}

Cele care au adus cel mai bun profit:

{'Jocurile Foamei', 'Death Note Vol 8', 'Povestea Slujitoarei', 'Kafka pe malul marii', 'Harry Potter si piatra filosofala', 'One Piece Vol 10', 'Pride and Prejudice', '1984', 'Crime and Punishment', 'Harry Potter si camera secretelor'}

## b) Diferența

**Cod:**

top\_10\_vanzari.difference(top\_10\_profit)

**Output:**

{'A Clash of Kings', 'A Storm of Swords',

'A Tale of Two Cities', 'Wuthering Heights'}

## c) Reuniune

**Cod:**

top\_10\_vanzari.union(top\_10\_profit)

**Output:**

{'1984',

'A Clash of Kings',

'A Storm of Swords',

'A Tale of Two Cities',

'Crime and Punishment',

'Death Note Vol 8',

'Harry Potter si camera secretelor',

'Harry Potter si piatra filosofala',

'Jocurile Foamei',

'Kafka pe malul marii',

'One Piece Vol 10',

'Povestea Slujitoarei',

'Pride and Prejudice',

'Wuthering Heights'}

## d) Intersecție

**Cod:**

top\_10\_vanzari.intersection(top\_10\_profit)

**Output:**

{'1984',

'Crime and Punishment',

'Harry Potter si camera secretelor',

'Harry Potter si piatra filosofala',

'Jocurile Foamei',

'Pride and Prejudice'}

## e) Diferență simetrică (cărțile care nu sunt comune celor doua seturi)

**Cod:**

top\_10\_vanzari.symmetric\_difference(top\_10\_profit)

**Output:**

{'A Clash of Kings', 'A Storm of Swords', 'A Tale of Two Cities', 'Death Note Vol 8',

'Kafka pe malul marii', 'One Piece Vol 10', 'Povestea Slujitoarei', 'Wuthering Heights'}

# 5) Utilizarea dicționarelor

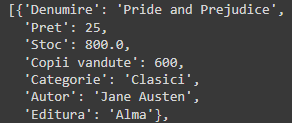
## a) Să se creeze o listă de dicționare pe baza coloanelor indexurilor coloanelor: 1, 2, 3, 5, 6, 7, 8

**Cod:**

dictVanzari = df.iloc[:, [1, 2, 3, 5, 6, 7, 8]].to\_dict(orient='records')

dictVanzari

**Output (parțial):**



## b) Să se creeze o funcție care calculează profitul generat de fiecare carte din lista de dicționare și să se afișeze acesta.

**Cod**:

def CalculProfit(stoc, pret, copiiVandute):

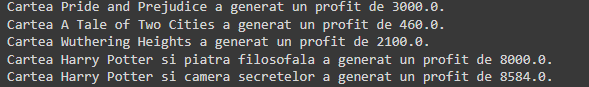
return (pret \* copiiVandute) - (0.6 \* pret \* stoc)

for carte in dictVanzari:

carte['Profit'] = CalculProfit(carte['Stoc'], carte['Pret'], carte['Copii vandute'])

print(f"Cartea {carte['Denumire']} a generat un profit de {carte['Profit']}.")

**Output (parțial):**



## c) Folosind metodele specifice dicționarelor să se afișeze cărțile scrise de fiecare autor.

**Cod:**

def Bibliografie(autor):

print(f'Cărțile scrise de {autor} sunt următoarele: ')

for carte in dictVanzari:

if 'Autor' in carte.keys() and carte.get('Autor') == autor:

print(carte.get('Denumire'))

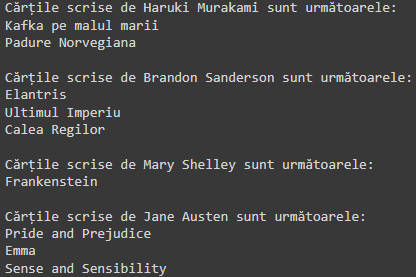
print('')

numeAutori = set([carte.get('Autor') for carte in dictVanzari])

for autor in numeAutori:

Bibliografie(autor)

**Output (parțial):**



## d) Să se elimine din lista de dicționare toate cărțile care au înregistrat un profit negativ:

cartiEliminate = []

for i in range(len(dictVanzari)):

if dictVanzari[i]['Profit'] < 0:

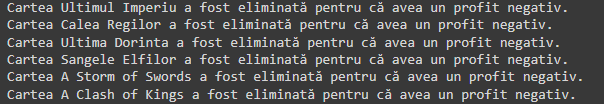
cartiEliminate.append(dictVanzari[i])

print(f"Cartea {dictVanzari[i]['Denumire']} a fost eliminată pentru că avea un profit negativ.")

for carte in cartiEliminate:

dictVanzari.pop(dictVanzari.index(carte))

**Output:**

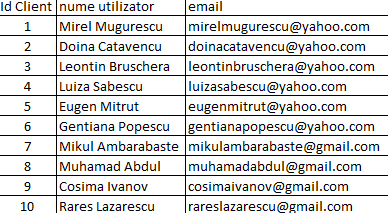


# 6) Folosirea merge-ului, grupării și vizualizarea datelor.

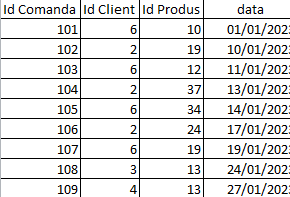
**Descrierea problemei:**

Firma dorește să se facă legătura între seturile de date corespunzătoare vânzărilor, comenzilor și clienților, apoi să se realizeze grupări care să permită realizarea unor grafice relevante.

**Informații necesare pentru rezolvare**



**Figura 6: Conținutul csv-ului date\_utilizatori**



**Figura 7: O parte din conținutul csv-ului date\_comenzi**

**Rezolvarea cu ajutorul produsului software**

Pentru început a fost realizat importul datelor:

utilizatori = pd.read\_csv('date\_utilizatori.csv', encoding='ISO-8859-2')

comenzi = pd.read\_csv('date\_comenzi.csv', encoding='ISO-8859-2')

Apoi a fost realizat merge-ul:

mergeInitial = pd.merge(comenzi, df, on='Id Produs', how='inner')

mergeFinal = pd.merge(mergeInitial, utilizatori, on='Id Client', how='inner')

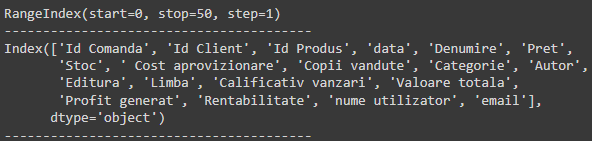
mergeFinal.to\_csv('date\_comerciale.csv', index=False)

print(mergeFinal.index)

print('-'\*40)

print(mergeFinal.columns)

print('-'\*40)



Am realizat prima grupare:

numarComenzi = mergeFinal.groupby('nume utilizator')['Id Comanda'].count().reset\_index().rename(columns={'nume utilizator': 'Nume Utilizator', 'Id Comanda': 'Numar Comenzi'})

numarComenzi



**Figura 8: Număr Comenzi clienti**

Pe baza acestei grupări, urmând să realizez următorul grafic:

plt.figure(figsize=(10, 6))

plt.bar(numarComenzi['Nume Utilizator'], numarComenzi['Numar Comenzi'])

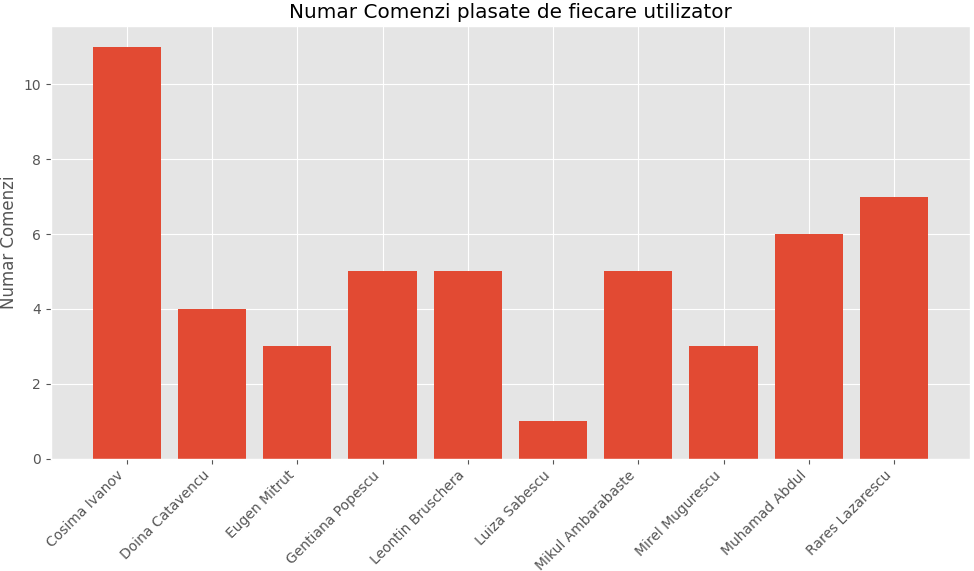
plt.ylabel('Numar Comenzi')

plt.title('Numar Comenzi plasate de fiecare utilizator')

plt.xticks(rotation=45, ha='right')

plt.tight\_layout()

plt.show()



**Figura 9: Frecvența comenzilor plasate de fiecare utilizator**

Cea de-a doua grupare realizată în cadru cerinței este următoarea:

mergeFinal['Categorie'].replace('DIstopic', 'Distopic', inplace=True)

distributieCategorii = mergeFinal.groupby(['nume utilizator','Categorie'])['Id Comanda'].count()

distributieCategorii



**Figura 9: repartiția pe categorie a comenzior clienților**

Pe baza acestei grupări am realizat un plot care să cuprindă un pie chart al preferințelor utilizatorilor care au comandat cărți din cel puțin 2 categorii diferite:

plt.rcdefaults()

plt.style.use('ggplot')

usersMultipleCategories = distributieCategorii.groupby('nume utilizator').filter(lambda x: len(x) > 1).index.get\_level\_values('nume utilizator').unique()

numUsers = len(usersMultipleCategories)

numCols = 3

numRows = (numUsers + numCols - 1) // numCols

fig, axes = plt.subplots(numRows, numCols, figsize=(15, 5\*numRows))

fig.set\_facecolor('lightgrey')

axes = np.array(axes).reshape(-1)

for i, user in enumerate(usersMultipleCategories):

userData = distributieCategorii[user]

ax = axes[i]

ax.pie(userData, labels=userData.index, autopct='%1.1f%%')

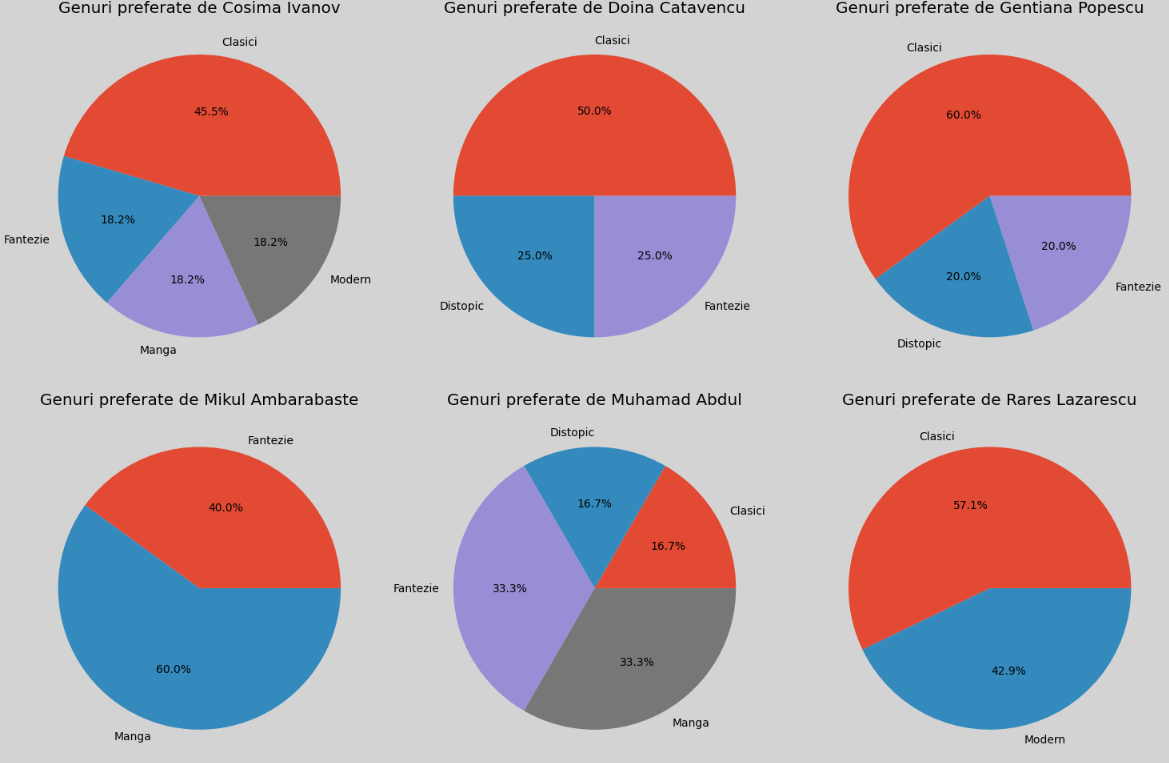
ax.set\_title(f"Genuri preferate de {user}")

for j in range(i+1, len(axes)):

axes[j].axis('off')

plt.tight\_layout()

plt.show()



**Figura 10: Piechart Preferințe ale clienților**

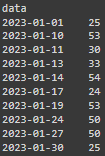
Gruparea finală pe care am realizat-o în cadrul acestei cerințe este următoarea:

evolutieVanzari = mergeFinal.groupby('data')['Pret'].sum()

evolutieVanzari.index = pd.to\_datetime(evolutieVanzari.index, format='%d/%m/%Y')

evolutieVanzari = evolutieVanzari.sort\_index()

evolutieVanzari



**Figura 11: Grupare venit pe dată**

Pe baza acestei ultime grupări am realizate 2 grafice:

**I)**

plt.figure(figsize=(14, 6))

evolutieVanzari.index = pd.to\_datetime(evolutieVanzari.index, format='%d/%m/%Y').strftime('%m-%d')

plt.plot(evolutieVanzari.index, evolutieVanzari.values, marker='o', linestyle='-')

plt.title('Trendul Venitului Zilnic')

plt.xlabel('Data')

plt.ylabel('Venitul Zilnic')

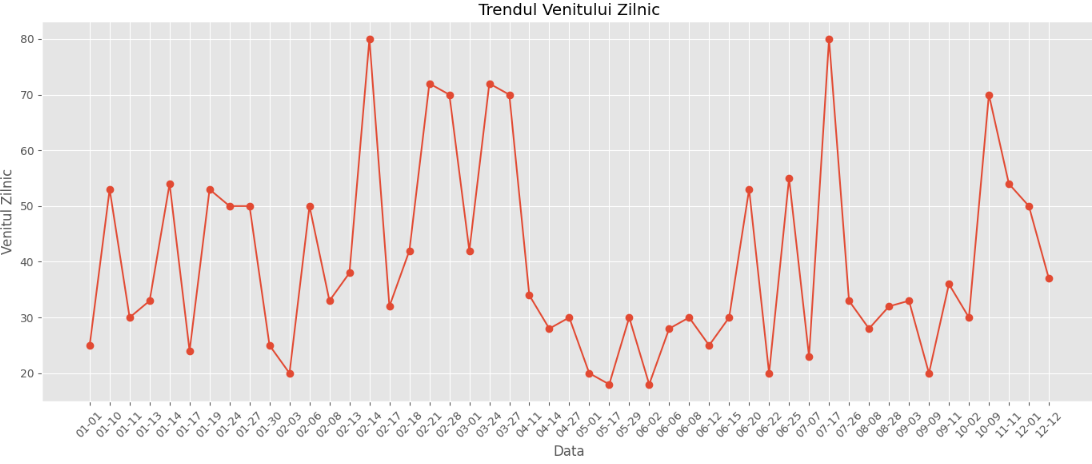
plt.xticks(rotation=45)

plt.grid(True)

plt.tight\_layout()

plt.show()

print(evolutieVanzari.index)



**Figura 12: Trendul zilnic al venitului**

În figură sunt surprinse valorile înregistrate în fiecare dată în care a fost înregistrată o comandă de către un utilizator. Pentru a oferi claritate sportiă am convertit formatul datei de la DD.MM.YYYY la MM.DD.

**II)**

cumulativeIncome = evolutieVanzari.cumsum()

plt.figure(figsize=(14, 6))

plt.plot(cumulativeIncome.index, cumulativeIncome.values, marker='o', linestyle='-')

plt.title('Evolutia venitului din vanzari')

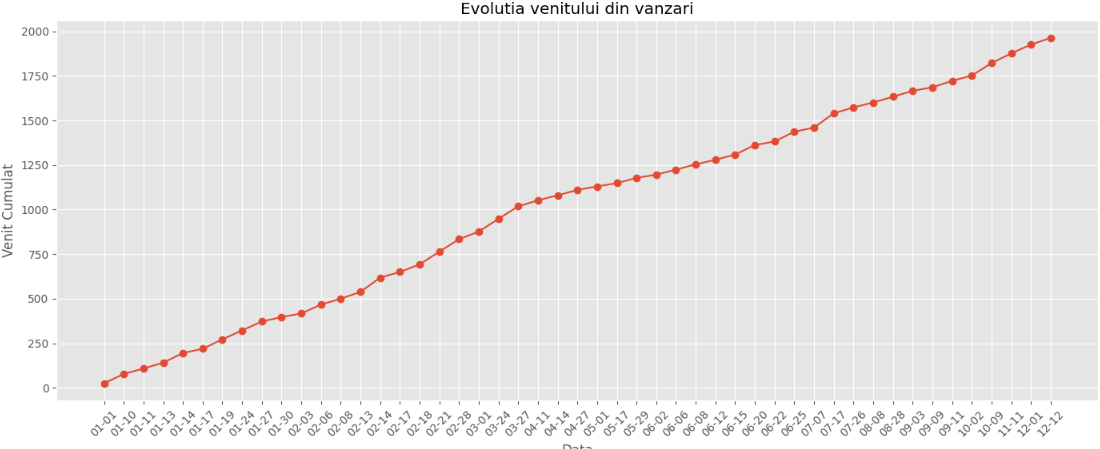
plt.xlabel('Data')

plt.ylabel('Venit Cumulat')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()



**Figura 13: Evoluția venitului din comenzi**

În cadrul acestui grafic este surprinsă creștere cumulativă a venitului de la data inițială la ultima surprinsă în csv-ul comenzi.