ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

Διαχείριση Σύνθετων Δεδομένων

Assignment1

Φωτόπουλος Στέφανος ,4829

Ιωάννινα,19/03/2023

Contents

1.Περιγραφή της εργασίας

Μέρος 1

Group-By With Aggregation:

```python
# Stefanos Fotopoulos 4829
import sys
import csv
from collections import defaultdict


def merge_sort(array, attribute_index):
    if len(array) <= 1:
        return array

    mid = len(array) // 2  # xwrizw ton pinika se 2 mikroterous
    first_half = merge_sort(array[:mid], attribute_index)#anadromiki klisi tis sinartisis mexri na exei mono 1 stoixeio kratontas mono to attribute pou mas endiaferei
    second_half = merge_sort(array[mid:], attribute_index)#anadromiki klisi tis sinartisis mexri na exei mono 1 stoixeio kratontas mono to attribute pou mas endiaferei
    sorted_array = []
    first_half_index = 0
    second_half_index = 0

    while first_half_index < len(first_half) and second_half_index < len(second_half):
        if int(first_half[first_half_index][attribute_index]) < int(second_half[second_half_index][attribute_index]): #elegxw poia timi einai megalyteri apo tis 2 gia na tin eisagw ston n
            sorted_array.append(first_half[first_half_index])
            first_half_index += 1
        else:
            sorted_array.append(second_half[second_half_index])
            second_half_index += 1

    sorted_array.extend(first_half[first_half_index:])#edw eisagoume ta enapomeinanta stoixeia ta opoia den mpikan ston sorted array
    sorted_array.extend(second_half[second_half_index:])#edw eisagoume ta enapomeinanta stoixeia ta opoia den mpikan ston sorted array
    return sorted_array
```

Στη merge_sort ελέγχω κάθε φορά στην αρχή κατά πόσο το μήκος του πίνακά μου έχει φτάσει τη μονάδα καθώς αφού τον χωρίσω στη μέση σε δυο υποπίνακες θα την καλέσουν αναδρομικά. Με αυτή την αναδρομή επιτυγχάνω να σπάσω σε μικρότερα κομμάτια τον αρχικό μου πίνακα, έτσι ώστε να γίνει ευκολότερη η ταξινόμισή του. Αφού τελειώσω με τις αναδρομές θα χρειαστεί να ανασυνθέσω τον παλιό μου πίνακα για αυτό τον λόγο υπάρχει η while. Καθώς σε κάθε επανάληψη κοιτάει ποια από τις δυο τιμές των νέων πινάκων είναι μεγαλύτερη για να την εισάγει στον sorted μεγαλώνοντας το index αυτού του πίνακα κατά ένα για να ελέγξω την επόμενη τιμή του. Τέλος, υπάρχει περίπτωση να μείνουν κάποια στοιχεία που δεν έχουν εισαχθεί στον sorted_array για αυτό χρησιμοποιώ τα extend.

```python
def group_by_aggregation(sorted_data, group_by_attr, aggregation_attr, function):
    data = defaultdict(list)

    for row in sorted_data:#diatrexei kathe grammi tou sorted data pou phre apo tin merge sort
        group_by = row[group_by_attr]
        aggregation_value = row[aggregation_attr]
        data[group_by].append(aggregation_value)#prosthetei tin aggregation value sto leksiko analoga me to ekastote group by

    with open('O1.csv', 'w', newline='') as output:
        writer = csv.writer(output)
        for group_by, aggregation_values in data.items():#anatrexei gia to sygkekrimeno kleidi(group_by) stis times(aggregation_values) pou exei to leksiko
            if function == 'sum':
                result = sum(map(int, aggregation_values))#xreiazetai map gia na paroume ton akeraio
            elif function == 'min':
                result = min(map(int, aggregation_values))#xreiazetai map gia na paroume ton akeraio
            elif function == 'max':
                result = max(map(int, aggregation_values))#xreiazetai map gia na paroume ton akeraio
            writer.writerow([group_by, result])

```

Για την group_by_aggregation δημιουργώ ένα λεξικό που θα με βοηθήσει να βρω τις τιμές για κάθε group_by_attribute.Αρχικά, διατρέχω τα δεδομένα που έγιναν sort από την προηγούμενη συνάρτηση και αρχικοποιώ τις μεταβλητές μου. Προσθέτω στο λεξικό μου την τιμή για το εκάστοτε κλειδί σύμφωνα με το group_by.Δημιουργώ ένα νέο αρχείο για εγγραφή όπως και έναν writer για αυτό το αρχείο. Με μια for loop ανατρέχω στο λεξικό μου για κάθε κλειδί της group_by και ανάλογα με το επιθυμητό Function δίνω το αποτέλεσμα το οποίο γράφω στο νέο αρχείο δίπλα από το group_by_attribute.

```python
if __name__ == "__main__":
    if len(sys.argv) != 5:
        print("Warning follow the pattern: python program.py <filename> <group_by_attribute> <aggregation_attribute> <function>")
        sys.exit(1)

    filename = sys.argv[1]
    group_by_attribute = int(sys.argv[2])
    aggregation_attribute = int(sys.argv[3])
    aggregation_function = sys.argv[4]

    with open(filename, newline='') as file:
        reader = csv.reader(file)
        data = list(reader)

    sorted_data = merge_sort(data, group_by_attribute)

    group_by_aggregation(sorted_data, group_by_attribute, aggregation_attribute, aggregation_function)

```

Στην main δείχνω στον χρήστη ποια είναι η σωστή σειρά των arguments που πρέπει να γράψει στην κονσόλα για να τρέξει το πρόγραμμα. Έπειτα, ανοίγω το αρχείο μου δημιουργώ έναν αναγνώστη για το αρχείο αυτό και διαβάζω τα δεδομένα από το αρχείο ως λίστες. Έτσι πλέον έχω μια λίστα από λίστες, όπου κάθε λίστα περιέχει τα δεδομένα από μια γραμμή του αρχικού csv αρχείου.
!!!!Προσοχή: Παρατήρησα ότι για να αλλάξουν τα δεδομένα στο νέο αρχείο θα χρειαστεί πρώτα να κλείσετε το παλιό(O1.csv) δηλαδή αν τρέξετε το αρχείο για max το ανοίξετε στο Excel και ενώ το έχετε ανοιχτό τρέξετε το πρόγραμμα για min θα σας πετάξει PermissionError: [Errno 13] Permission denied: 'O1.csv'. Εάν κλείσετε το αρχείο(O1.csv για το Max) θα αλλάξει το O1.csv και θα δείτε το O1.csv με τις αλλαγμένες πλέον τιμές για min.

Μέρος 2

Merge join:

```python
#Stefanos Fotopoulos 4829
import csv

def merge_join(R_csv, S_csv, output):
    with open(R_csv, newline='') as R, open(S_csv, newline='') as S, open(output, 'w', newline='') as output_csv:
        #Dimiourgia readers/writer gia na mporesw na diatreksw to arxeio kai na grapsw ta apotelesmata sto output
        reader_R = csv.reader(R)
        reader_S = csv.reader(S)
        writer = csv.writer(output_csv)

        for row_R in reader_R:
            S.seek(0) #Epanaferw ton file pointer stin arxi tou S
            for row_S in reader_S:
                if row_R[0] == row_S[1]: #R(ABC) S(DAE) elegxw tin prwth stili tou R me tin mesaia tou S
                    writer.writerow(row_R + row_S[:1] + row_S[2:]) #Krataw mono ABC apo R kai DE apo S

if __name__ == "__main__":
    merge_join('R.csv', 'S.csv', '02.csv')

```

Με την βοήθεια της with ανοίγω τα αρχεία .csv για ανάγνωση και εγγραφή και δημιουργώ readers και writer έτσι ώστε να διατρέξω το αρχείο και να γράψω μέσω του writer τα αποτελέσματα στο αρχείο output.Με το for loop που έπεται διατρέχω την κάθε γραμμή της πρώτης στήλης του R με τις εγγραφές της μεσαίας στήλης του S και εάν αυτές οι τιμές είναι ίσες κρατάω το αποτέλεσμα(ABCDE) στο αρχείο output.Τέλος, θα χρειαστεί στην αρχή κάθε ‘τρεξίματος’ να επαναφέρουμε τον file pointer του S στο 0 για να συγκριθεί με την επόμενη τιμή του R.

Μέρος 3

Composite Query:

```python
#Stefanos Fotopoulos 4829
import csv

def composite_query(R_csv, S_csv, output):

    with open(R_csv, newline='') as R, open(S_csv, newline='') as S, open(output, 'w', newline='') as output_csv:
        # Dimiourgia readers/writer gia na mporesw na diatreksw to arxeio kai na grapsw ta apotelesmata sto output
        reader_S = csv.reader(S)
        reader_R = csv.reader(R)
        writer = csv.writer(output_csv)

        for row_R in reader_R:
            R_A = row_R[0]
            C = int(row_R[2])
            sum = 0

            if C == 7:
                S.seek(0)  # Epanaferw ton file pointer stin arxi tou S
                for row_S in reader_S:
                    S_A = row_S[1]
                    E = int(row_S[2])
                    if R_A == S_A:
                        sum += E
                if sum > 0: # Xwris auth thn synthiki krataei kai tis times pou exoun mono R.C = 7
                    writer.writerow([R_A, sum])


if __name__ == "__main__":
    composite_query('R.csv', 'S.csv', '03.csv')

```

Με την βοήθεια της with ανοίγω τα αρχεία .csv για ανάγνωση και εγγραφή και δημιουργώ readers και writer έτσι ώστε να διατρέξω το αρχείο και να γράψω μέσω του writer τα αποτελέσματα στο αρχείο output.Αρχικοποιώ τα R_A,C,sum(μεταβλητή στην οποία θα κρατήσω το άθροισμα) σύμφωνα με τα δεδομένα της εκφώνησης δηλαδή το R_A είναι η 1η στήλη του R και το C η 3η όπου συγκρατώ τον integer.Ελέγχω πρώτα για το αν η τιμή του C είναι 7 και εάν ισχύει δίνω τιμές στα S_A και E. Τέλος, όταν ισχύει η ισότητα προσθέτω στο sum την τιμή του E, θα χρειαστεί στην αρχή κάθε ‘τρεξίματος’ να επαναφέρουμε τον file pointer του S στο 0 για να συγκριθεί με την επόμενη τιμή του R.