בס"ד

חלק א'

סעיף א':

5. סיבוכיות הזמן של הפתרון: (m log m)O כאשר m הוא גודל הקובץ.

הסבר:

O(m): מעבר על כל הזוגות במפה.

O(m log m): מיון המפה.

O(n): לקיחת n האיברים הראשונים. (לא משפיע על הסיבוכיות)

סיבוכיות מקום: O(m) מפה שמכילה את כל ערכי הקובץ.

סעיף ב':

3. כדי לתכנן פתרון עבור נתונים שמגיעים בזרימה (stream), יש מספר גישות טכניות שניתן לשקול:

1. להשתמש בכלים כמו Apache Kafka, Apache Flink או Apache Spark Streaming כדי לעבד את הנתונים בזמן אמת. כלים אלו מאפשרים לקבל, לעבד ולשלוח נתונים בזרימה בצורה יעילה.
2. להשתמש במסדי נתונים NoSQL כמו MongoDB או Cassandra, אשר מתאימים יותר לזרימה של נתונים בזמן אמת
3. להשתמש במערכת תורים כמו RabbitMQ או AWS SQS כדי לנהל את הנתונים המגיעים בזרימה. זה מאפשר לנהל את העומס ולהבטיח שהנתונים מעובדים בצורה מסודרת.
4. לתכנן את המערכת כקבוצת מיקרו-שירותים, כאשר כל שירות אחראי על חלק מהעיבוד של הנתונים. זה מאפשר גמישות והרחבה קלה של המערכת.
5. לתכנן מנגנוני טיפול בשגיאות כדי להתמודד עם בעיות שעשויות להתרחש במהלך עיבוד הנתונים. זה יכול לכלול חזרה על עיבוד נתונים או שמירה על נתונים לא מעובדים לצורך עיבוד מאוחר יותר.

**.4 יתרונות מרכזיים של פורמט Parquet:**

**1.  פורמט עמודתי (Columnar Format)**

Parquet שומר את הנתונים לפי עמודות (ולא לפי שורות כמו CSV), מה שמאפשר:

* קריאה סלקטיבית של עמודות (ולא את כל הנתונים)
* ביצועים גבוהים בעיבודים אנליטיים

**2.  חסכון בנפח בזכות דחיסה חכמה**

Parquet דוחס כל עמודה בנפרד – מה שמוביל ל:

* **קבצים קטנים משמעותית** מ־CSV
* **מהירות העברה גבוהה יותר** ברשת
* **יעילות אחסון בענן** (S3, GCS וכו'

**3.  ביצועים גבוהים בעיבוד נתונים**

* קריאה של מיליוני שורות בפחות זיכרון
* תומך ב־lazy loading, projection, predicate pushdown
* אידיאלי לעבודה עם Apache Spark, Presto, Hive, AWS Athena

**4.  סכמות נתונים ברורות (Typed Schema)**

* כל עמודה בפורמט Parquet שומרת טיפוס נתונים ברור (DateTime, Double, Integer…)
* אין צורך לנחש טיפוסים כמו ב־CSV
* אפשר לעשות ולידציה אוטומטית וטעינה טיפוסית (typed deserialization)

**5.  תמיכה מצוינת בכלים ותשתיות מודרניות**

* תואם ל־Pandas, PySpark, Hive, DuckDB, Trino, AWS Athena, GCP BigQuery
* אפשר לכתוב ולקרוא אותו מכל שפת תכנות מודרנית

**6.  מותאם לעבודה בענן ו־Data Lake**

* משתלב היטב עם תצורת Lakehouse
* נפוץ מאוד באחסון ארוך טווח של מידע גולמי
* תומך בחיפושים מהירים על כמויות עצומות של נתונים