Projektni zadatak za ocenu 10+ iz predmeta

Napredni algoritmi i strukture podataka

Školska godina 2023/2024.

1. Vremenske serije

Vremenske serije predstavljaju uređeni niz vrednosti neke promenljive zabeleženih najčešće u jednakim vremenskim intervalima (na primer na svakih sat vremena). Dakle, one predstavljaju niz podataka u diskretnim vremenskim tačkama. Na primer, podaci sa vremenskim oznakama, poput log fajlova, metrika aplikacije i sistema ili merenja sa IoT uređaja, mogu se smatrati vremenskim serijama. Pored toga, primer vremenskih serija mogu biti merenja temperature na određenoj meteorološkoj stanici, dnevne vrednosti cena određenih akcija i slično.

Podaci koji se klasifikuju kao vremenske serije poseduju nekoliko osobina koje ih izdvajaju od drugih tipova podataka:

- **Sekvencijalnost**: Podaci u vremenskim serijama su zabeleženi hronološkim redosledom, što omogućava praćenje promena kroz vreme i identifikaciju uzročno-posledičnih odnosa.
- **Nepromenljivost**: Svaka tačka u vremenskoj seriji je zapis nekog događaja i, kada je jednom zapisana, ne očekuje se da će se kasnije menjati.
- **Vremenski intervali**: Podaci se najčešće (ali ne uvek) prikupljaju u pravilnim vremenskim razmacima (sekunde, minuti, sati, dani, godine), što omogućava konzistentnu analizu.
- **Visoka frekvencija podataka**: Merenja se često vrše u kratkim intervalima (svake sekunde ili manje), što je uobičajeno kod IoT uređaja i finansijskih tržišta.
- **Multivarijantnost**: Vremenske serije mogu uključivati više promenljivih koje se beleže istovremeno, npr. temperatura, vlažnost i pritisak na istom mestu.

2. Model podataka

Storage engine specijalizovan za rad sa vremenskim serijama treba da podrži model podataka prilagođen njegovoj nameni. Kroz projekat treba da podržite sledeći model:

Jednu **vremensku seriju** jedinstveno identifikuju:

- **Naziv merenja**: Na primer temperatura, procenat osvetljenja, vrednost akcije itd.
- **Skup tagova**: Svaki tag se sastoji iz naziva i vrednosti i služi da bliže odredi vremensku seriju. Na primer, u slučaju gde merimo temperaturu na više lokacija, temperatura na svakoj lokaciji predstavlja po jednu vremensku seriju. Kako bismo ih mogli identifikovati, uz svaku možemo pridružiti tag sa nazivom *lokacija*. Tada bi prva vremenska serija za sebe imala vezan tag *lokacija=Šid*, druga *lokacija=Beograd*, itd.

Jednu **tačku (ili vrednost)** jedinstveno identifikuju:

- **Identifikator vremenske oznake**: Na primer
  - naziv merenja: temperatura
  - tagovi: lokacija=Novi Sad
- **Vremenska oznaka**: Recimo 2024-12-04T15:15:32

Naziv merenja, naziv i vrednost taga treba da budu tipa **string**, dok vrednost merenja treba da bude tipa **float**.

Kroz tabele koje se nalaze ispod ovog teksta možemo videti primer podataka upisanih u storage engine:

		Tagovi		
Naziv merenja	**Vremenska oznaka**	**Lokacija**	**ID senzora**	**Vrednost**
Temperatura	2024-12-04T15:14:32	Novi Sad	aa:bb:cf:d9:2a:12	12.3
Temperatura	2024-12-04T15:14:32	Beograd	ba:bb:cf:d9:2a:12	10.2
Temperatura	2024-12-04T15:15:32	Novi Sad	aa:bb:cf:d9:2a:12	12.4
Temperatura	2024-12-04T15:15:32	Beograd	ca:bb:cf:d9:2a:12	10.0

		Tagovi		
Naziv merenja	**Vremenska oznaka**	**Lokacija**	**ID senzora**	**Vrednost**
Vlažnost vazduha	2024-12-04T15:14:32	Novi Sad	aa:bb:cf:d9:2a:12	82.4
Vlažnost vazduha	2024-12-04T15:14:32	Beograd	ba:bb:cf:d9:2a:12	90.1
Vlažnost vazduha	2024-12-04T15:15:32	Novi Sad	aa:bb:cf:d9:2a:12	81.9
Vlažnost vazduha	2024-12-04T15:15:32	Beograd	ba:bb:cf:d9:2a:12	90.3

Iz podataka možemo identifikovati pet vremenskih serija:

- **Prva**
  - **naziv merenja**: Temperatura
  - **tagovi**: lokacija=Novi Sad, ID senzora=aa:bb:cf:d9:2a:12
- **Druga**
  - **naziv merenja**: Temperatura
  - **tagovi**: lokacija=Beograd, ID senzora=ba:bb:cf:d9:2a:12
- **Treća**
  - **naziv merenja**: Temperatura
  - **tagovi**: lokacija=Beograd, ID senzora=ca:bb:cf:d9:2a:12
- **Četvrta**
  - **naziv merenja**: Vlažnost vazduha
  - **tagovi**: lokacija=Novi Sad, ID senzora=aa:bb:cf:d9:2a:12
- **Peta**
  - **naziv merenja**: Vlažnost vazduha
  - **tagovi**: lokacija=Beograd, ID senzora=ba:bb:cf:d9:2a:12

3. API

Engine treba da podrži sledeće operacije:

- **WRITE_POINT -** zapis nove tačke/vrednosti
    - **measurement_name**: string
    - **tags**: map[string]string
    - **timestamp**: int
    - **value**: float
- **DELETE_RANGE -** brisanje dela serije koji je u vremenskom opsegu
    - **measurement_name**: string
    - **tags**: map[string]string
    - **min_timestamp**: int
    - **max_timestamp**: int
- **LIST** - dobavljanje serija u navedenom intervalu
    - **measurement_name**: string
    - **tags**: map[string]string
    - **min_timestamp**: int
    - **max_timestamp**: int
- **AGGREGATE** - primena agregacione funkcije nad navedenom serijom
    - **measurement_name**: string
    - **tags**: map[string]string
    - **min_timestamp**: int
    - **max_timestamp**: int
    - **aggregation_func: min | max | mean | avg**

4. Nefunkcionalni zahtevi

4.1 Engine treba da se sastoji iz struktura optimizovanih za rad sa vremenskim serijama

4.2 Format podataka na disku možete definisati sami uz oslonac na format koji ste implementirali za regularan projekat ili možete koristiti Parquet[1] format koji biste u tom slučaju implementirali samostalno od nule.

4.3 Potrebno je, gde je adekvatno, primeniti tehnike kompresije podataka kao što su delta, delta of delta, dictionary kompresiju i ostale tehnike koje ćete istražiti.

4.4 Engine treba da omogući korisniku da definište **retention period**. Retention period je vremenski period nakon kog podaci treba da se obrišu. Recimo, ako je retention period 7 dana, sve tačke starije od toga treba tretirati kao obrisane.

4.5 Duplikate i zastarele podatke potrebno je uklanjati procesom kompakcija.

[1] https://parquet.apache.org/

5. Pomoćni materijali

- https://docs.influxdata.com/influxdb/v2/reference/internals/storage-engine/
- https://docs.influxdata.com/influxdb/v2/reference/internals/file-system-layout/
- https://docs.influxdata.com/influxdb/v2/reference/internals/data-retention/
- https://www.influxdata.com/blog/compactor-hidden-engine-database-performance/
- https://prometheus.io/docs/prometheus/latest/storage/
- https://github.com/prometheus/prometheus/tree/main/tsdb
- https://ganeshvernekar.com/blog/prometheus-tsdb-the-head-block/
- https://tdengine.com/storage-engine-comparison-between-tdengine-and-prometheus/
- http://www-cs-students.stanford.edu/~adityagp/courses/cs598/papers/dremel.pdf