Projektmunka 3 dokumentáció

Balogh Bence EABZRE

1. feladat

A prezentáció alapján elkészítettem a Producert.
Amire figyelnem kellet:

- a konfigban néhány adat eltér:
  - topic, username, password
  - servers (mivel csak a Google szervere volt a hely, a többire azt írta, hogy már nincs hely)
  - hozzáadtam új adatok, amelyek a marketCapUsd és a vwap24Hr

Az új adatokhoz a következő módosításokat kellett elvégezni:

Producer:
A json feldolgozáskor ezeket az adatokat is fel kell dolgozni.

```
while True:
    try:
        response = requests.get('https://api.coincap.io/v2/assets?limit=20')
        #print(response)
        if response.status_code == 200:
            for coin in pd.DataFrame.from_dict(response.json()['data']).loc[:,['symbol','priceUsd','changePercent24Hr','marketCapUsd','vwap24Hr']].values.tolist():
              #print(coin)
              p.produce(topic,key = "key", value = coin[0] + ", " + coin[1] + ", " + coin[2])
    except:
      pass
    print("produce... (every 10 seconds)")
    sleep(10)
```

Consumer:
Ebben a projektben semmit se kellett módosítani.

DB_process:
Az adatbázis táblában felvesszük az új adatok sorait.

```
%sql
CREATE TABLE cointable (
coin STRING,
price Double,
difference Double,
markcap Double,
vwap Double
)
USING delta
PARTITIONED BY (coin)
LOCATION '/mnt/delta/cointable'
```

OK

Command took 6.20 seconds -- by a user at 2021. 10. 10. 10:39:10 on unknown cluster

Cmd 2

```
import org.apache.spark.sql._
import org.apache.spark.sql.functions._
import org.apache.spark.sql.streaming._
import org.apache.spark.sql.types._

import spark.implicits._
case class DeviceData(coin: String,price: Double, difference: Double, markcap: Double, vwap: Double)
```

A DeviceDataFrame készítésnél hozzáfűzzük az új adatokat is és a classhez is hozzáadjuk ezeket.

```
val dfff = dff.map(line =>DeviceData(line(0),line(1).toDouble,line(2).toDouble,line(3).toDouble,line(4).toDouble))
```

2. feladat

A megadott minta alapján alapján elkészítettem kódokat
Az új adatoknál nagyjából ugyanazokat a változtatásokat kell elvégezni mint az első feladatban.
Amire figyelnem kellett:

- a konfigban néhány adat eltér:
    - topic, username, password
    - servers (mivel csak a Google szervere volt a hely, a többire azt írta, hogy már
    - hozzáadtam új adatok, amelyek a marketCapUsd és a vwap24Hr

Az új adatokhoz a következő módosításokat kellett elvégezni:

Producer:
A json feldolgozáskor ezeket az adatokat is fel kell dolgozni.

```
while True:
    try:
        response = requests.get('http s://api.coincap.io/v2/assets?limit=20')
        if response.status_code == 200:
            for coin in pd.DataFrame.from_dict(response.json()['data']).loc[:,['symbol','priceUsd','changePercent24Hr','marketCapUsd','vwap24Hr']].values.tolist():
              p.produce(topic,key = "key", value = coin[0] + ", " + coin[1] + ", " + coin[2] + ", " + coin[3] + ", " + coin[4])
    except:
      pass
    print("produce... (every 10 seconds)")
    sleep(10)
```

Running command...

Consumer:
Ebben a projektben semmit se kellett módosítani.
Realtime_priocess:
Itt sok minden módosítani kell.
Az osztályt

```
case class DeviceData(coin: String,price: Double, difference: Double, markcap: Double, vwap: Double)
```

A kulcs érték pár beállításnál a mapingnál.

```
val words = lines.map(splitted =>DeviceData(splitted.split(",")(0),splitted.split(",")(1).toDouble,splitted.split(",")(2).toDouble),splitted.split(",")(3).toDouble),splitted.split(",")(4).toDouble))
```

```
words: org.apache.spark.streaming.dstream.DStream[DeviceData] = org.apache.spark.streaming.dstream.MappedDStream@49566179
Command took 0.72 seconds -- by baloghbnc@stud.uni-obuda.hu at 2021. 12. 02. 10:33:53 on teszt
```

Cmd 6

```
val keyword = words.map(w=>(w.coin,(w.price,w.difference,w.markcap, w.vwap, 1)))
```

Valamint a kimenetben hozzá kell fűzni az új étrékeket is.

```
val output = final_data.map(w=>w.coin+","+w.price+","+w.difference+","+w.markcap+","+w.vwap)
```

3. feladat

A harmadik feladatban egy XML fájlt kell beolvasni majd avro fájlként menteni.

Első lépésként telepíteni kell a pandavro és a pandas csomagokat a pip segítségével.

```
!pip install pandavro
!pip install pandas
```

Utána be kell importálni a szükséges könyvtárakat. A beépített xml kezelt használjuk. A pandavro lesz a kapcsolat a dataframe és a avro fájl között, a pandas csomagból pedig magát a dataframet használjuk.

```
import xml.etree.ElementTree as et
import pandavro as pdx
import pandas as pd
```

Következő lépésben elkészítjük az XML feldolgozó metódust, amely egy dictionaryt ad vissza.

```
def readfromxmltodict(path):
    tree = et.parse(path)
    root = tree.getroot()

    data = {}
    i = 0
    for child in root:
        data[i] = []
        for ch in child:
            data[i].append(ch.text)
        i+=1
    return data
```

A metódus meghívása előtt - mivel nem töltöttem fel a gépre az XML fájlt - létre kell hoznom a fájlt, amelyet az urllib2 könyvtár urlopen függvényével hozom létre. Ez az adatokat fájl szerűen tárolja el így. Ezután meghívhatom a megírt metódusunkat.

```
file = urllib2.urlopen('https://www.w3schools.com/xml/books.xml')

dictData = readfromxmltodict(file)
```

Végül megadjuk a kimeneti fájl helyé, majd elkészítjük az avro fájlt és lementjük.

```
OUTPUT_PATH="output.avro"
pdx.to_avro(OUTPUT_PATH, pd.DataFrame.from_dict(dictData[0]))
saved = pdx.read_avro(OUTPUT_PATH)
print(saved)
```