DĚLAT
DOBRÝ SOFTWARE
NÁS BAVÍ

PROFINIT

Spark SQL, Spark Streaming

Jan Hučín

22. listopadu 2017

Osnova

1. Spark SQL
2. Další rozšíření Sparku
   - Spark streaming
   - GraphX
   - Spark ML

PROFINIT

Spark SQL

Spark SQL a DataFrames (DataSets)

- Rozšíření k tradičnímu RDD přístupu
- Datová struktura **DataFrame** = RDD se sloupci
  - obdoba databázové relační tabulky
  - obsahuje i schéma
  - nad rámec RDD – práce se sloupci
  - možnost použití syntaxe podobné SQL nebo přímo SQL

1;Andrea;35;64.3;Praha
2;Martin;43;87.1;Ostrava
3;Simona;18;57.8;Brno

id	jmeno	vek	hmotnost	mesto
1	Andrea	35	64.3	Praha
2	Martin	42	87.1	Ostrava
3	Simona	18	57.8	Brno

Spark SQL – výhody a nároky

- Výhody oproti tradičnímu Sparku (RDD):
  - stručnější a jednodušší kód
  - využití Hive
  - snazší optimalizace
  - ⇨ rychlejší běh

- Nároky navíc:
  - rozšířené API: objekt **`sqlContext`**, ev. další

- Kdy nelze použít?
  - úlohy nevhodné pro SQL ⇨ tradiční Spark
  - úlohy náročné na paměť ⇨ map-reduce, Hive

PROFINIT

Příklad – společné zadání

- Který stát USA má na meteostanicích nejvyšší průměrný normál v létě?
  (již jsme řešili pomocí Hive)

Struktura dat:

```
stanice,mesic,den,hodina,teplota,flag,latitude,longitude,vyska,stat,nazev
AQW00061705,1,1,1,804,P,-14.3306,-170.7136,3.7,AS,PAGO PAGO WSO AP
AQW00061705,1,2,1,804,P,-14.3306,-170.7136,3.7,AS,PAGO PAGO WSO AP
AQW00061705,1,3,1,803,P,-14.3306,-170.7136,3.7,AS,PAGO PAGO WSO AP
AQW00061705,1,4,1,802,P,-14.3306,-170.7136,3.7,AS,PAGO PAGO WSO AP
AQW00061705,1,5,1,802,P,-14.3306,-170.7136,3.7,AS,PAGO PAGO WSO AP
```

PROFINIT

Postup 0 (tradiční RDD)

```
tp_raw = sc.textFile('/user/pascepet/teplota-usa')

tp_raw = tp_raw.filter(lambda r:
    (r.split(',')[1] in set('678')) & (r.split(',')[4] != '')

tp = tp_raw.map(uprav_radek)

tp_st = tp.reduceByKey(soucty) \

    .map(lambda x: (x[0], x[1][0]/x[1][1])) \

    .sortBy(lambda y: y[1], False)

tp_st.take(1)
```

```
uprav_radek

AQW00061705,7,30,4,804,P,-14.3306,-170.7136,3.7,AS,PAGO PAGO WSO AP
                                  ↓
                          (AS, (26.89, 1))
```

PROFINIT

Jak vyrobit DataFrame?

- konverze existujícího RDD
  - převoditelné do sloupců
- načtení souboru
  - s již definovanými sloupci (např. Parquet, ORC)
  - převoditelné do sloupců (např. CSV)
- výsledek dotazu do Hive
- výsledek dotazu do jiné DB (JDBC konektor)

PROFINIT

Jak vyrobit DataFrame?

- konverze existujícího RDD
  - `sqlContext.createDataFrame(RDD, schema)`
- načtení souboru
  - `sqlContext.read.format(formát).load(cesta)`
- výsledek dotazu do Hive
  - `sqlContext.sql(dotaz_sql)`

PROFINIT

Postup 1 (CSV → RDD → DataFrame)

```
from pyspark.sql.types import *

tp_raw = sc.textFile('/user/pascepet/teplota-usa')

tp_raw = tp_raw.filter(lambda r:
    (r.split(',')[1] in set('678')) & (r.split(',')[4] != '')

tp_prep = tp_raw.map(uprav_radek_df)

tpDF = sqlContext.createDataFrame(tp_prep, tp_schema)
```

```
uprav_radek_df

AQW00061705,7,30,4,804,P,-14.3306,-170.7136,3.7,AS,PAGO PAGO WSO AP
                                  ↓
                              (AS, 26.89)
```

```
tp_pole = [StructField('stat', StringType(), True),
StructField('tepl', DoubleType(), True)]

tp_schema = StructType(tp_pole)
```

PROFINIT

Postup 2 (přímé načtení CSV → DataFrame)

```
tpDF2 = sqlContext.read \
        .format("com.databricks.spark.csv") \
        .option("header", "true") \
        .option("delimiter", ",") \
        .option("inferSchema", "true") \
        .load("/user/pascepet/teplota-usa")
```

Postup 3 (Hive → DataFrame)

```
tpDF3 = sqlContext.sql('select * from sstamenov.pocasi')
```

PROFINIT

Jak pracovat s DataFrame?

1. registrovat jako dočasnou tabulku + dotazování SQL
2. pseudo-SQL operace
3. operace RDD – výsledek může být jen obyčejné RDD

Jak pracovat s DataFrame?

1. registrovat jako dočasnou tabulku + dotazování SQL
   - `DF.registerTempTable("tabulka")`
   - `sqlContext.sql("select * from tabulka")`
2. pseudo-SQL operace
   - `DF.operace`, např. select, filter, join, groupBy, sort...
3. operace RDD – výsledek může být jen obyčejné RDD
   - např. map, flatMap...
   - řádek v DataFrame je typu **Row** – práce jako s typem **list**

PROFINIT

Pseudo-SQL a další operace

- **select** (omezení na uvedené sloupce)
- **filter** (omezení řádků podle podmínky)
- **join** (připojení jiného DataFrame)
- **groupBy** (seskupení)
- **agg, avg, count** (agregační funkce)
- **toDF** (přejmenování sloupců)
- **withColumn** (transformace sloupců)
- **show** (hezčí výpis obsahu DataFrame)

PROFINIT

Registrace dočasné tabulky

```
tpDF.registerTempTable("teploty")

tp_stDF = sqlContext.sql("""select stat, avg(tepl) as
tepl_prum from teploty
group by stat order by tepl_prum desc""")

tp_stDF.show(1)
```

```
tp_pole = [StructField('stat', StringType(), True),
StructField('tepl', DoubleType(), True),
StructField('mesic', DoubleType(), True)]

tp_schema = StructType(tp_pole)
```

PROFINIT

Pseudo-SQL

```
tpDF2 = tpDF2.filter((tpDF2.mesic>5) & (tpDF2.mesic<9)) \
    .select('stat','tepl').na.drop()
tpDF2 = tpDF2.withColumn('tepl', (tpDF2.tepl/10.0-32) * 5/9)
tpDF2 = tpDF2.groupBy('stat').avg() \
    .toDF('stat','prum')
tpDF2.sort(tpDF2.prum.desc()).limit(1).show()
```

PROFINIT

Spark Streaming

Co to je a jak to využít

- dávkové zpracování přicházejících dat
- příchozí data neklepou na dveře, sedí v čekárně
- near real-time, pevné nastavení časového okna

Možné využití:

- filtrování logů, zpráv
- monitorování, reakce na událost
- vyhledávání v nakešovaných datech

PROFINIT

Princip zpracování

- streamovací modul dávkuje příchozí data – posloupnost RDD
- klasický Spark postupně odbavuje RDD ve frontě
- API pro Javu, Scalu, s malým omezením i Python

PROFINIT

Příklad

Úkol: pro každou dávku ze socketu spočítat četnosti slov

```
sc = SparkContext(appName="Priklad")
ssc = StreamingContext(sc, 10)

lines = ssc.socketTextStream(sys.argv[1], int(sys.argv[2]))
counts = lines.flatMap(lambda line: line.split(" ")) \
                  .map(lambda word: (word, 1)) \
                  .reduceByKey(lambda a, b: a+b)

counts.pprint()

ssc.start()
ssc.awaitTermination()
```

PROFINIT

GraphX
Spark ML

GraphX

- rozšíření pro algoritmy prohledávající grafy
- ve stadiu vývoje
- připravené algoritmy:
  - PageRank
  - rozklad na podgrafy
  - počet trojúhelníků
  - label propagation
  - a další...
- API pro iterativní procházení grafů (Pregel)

PROFINIT

Spark ML (machine learning)

- klasické algoritmy machine learning, např.:
  - regrese, lineární modely
  - rozhodovací stromy
  - naivní Bayesův klasifikátor
  - shluková analýza
- algoritmy pro velká data, např.:
  - doporučovací systém
  - asociační pravidla, časté podmnožiny
- statistické metody, např.:
  - popisná statistika
  - testování hypotéz
- mnohorozměrné metody, např.:
  - hlavní komponenty
  - faktorová analýza

PROFINIT

Spark ML (machine learning)

- praktický smysl mají jen algoritmy pro velká data
- u ostatních metod:
  - z velkých dat se vybere vzorek
  - na vzorku se tradičními nástroji modeluje
  - navržený model se naprogramuje ve Sparku (bez Spark ML)

Díky za pozornost

PROFINIT

Profinit, s.r.o.
Tychonova 2, 160 00 Praha 6

Telefon
+ 420 224 316 016

Web
www.profinit.eu

LinkedIn
linkedin.com/company/profinit

Twitter
twitter.com/Profinit_EU