DĚLAT
DOBRÝ SOFTWARE
NÁS BAVÍ

PROFINIT

Spark SQL – cvičení

Jan Hučín

25. listopadu 2020

Opakování

- Základní datová struktura Sparku = RDD
- Rozšíření pro práci se sloupci = DataFrame
- DataFrame získáme:
  - konverzí RDD
  - přímým načtením souboru
  - z Hive

PROFINIT

Postup 1 (přímé načtení CSV → DataFrame)

```
DF1 = spark.read \
        .format("csv") \
        .option("header", "true") \
        .option("delimiter", ",") \
buď     .schema(nazev_schematu) \
nebo    .option("inferSchema", "true") \
        .load(cesta)
```

Postup 2 (Hive → DataFrame)

```
DF2 = spark.sql(SQL dotaz jako řetězec)
```

PROFINIT

Postup 3 (RDD → DataFrame)

- řádek → Row(pole1=hodnota, pole2=hodnota, ...)
- Row – něco jako *list*, umožňuje spojit data do sloupců
- CreateDataFrame si určí typy podle hodnot v Row
- můžeme ale při konverzi definovat i vlastní schéma
- místo CreateDataFrame lze použít i toDF

```
from pyspark.sql import Row
rdd_prep = tp_raw.map(transformace řádku na Row)
DF3 = spark.createDataFrame(rdd_prep)
```

PROFINIT

Jak pracovat s DataFrame?

1. registrovat jako dočasnou tabulku + dotazování SQL
   – výsledek je opět DataFrame
2. pseudo-SQL operace (podobné *pandas*)
   – výsledek je opět DataFrame
3. tradiční operace pro RDD (např. map, flatMap...)
   – nutno explicitně převést na obyčejné RDD

PROFINIT

Jak pracovat s DataFrame?

1. registrovat jako dočasnou tabulku + dotazování SQL
   - `DF.registerTempTable("tabulka")`
   - `spark.sql(SQL dotaz)`
2. pseudo-SQL operace
   - `DF.operace`, např. select, filter, join, groupBy, sort...
3. operace RDD – nutno explicitně převést na obyčejné RDD
   - `DF.rdd.map(...)` apod.
   - řádek v DataFrame je typu **Row** – práce jako s typem **list**

PROFINIT

Pseudo-SQL a další operace

Syntaxe i funkčnost je velmi podobná *pandas* pro Python.

- **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)

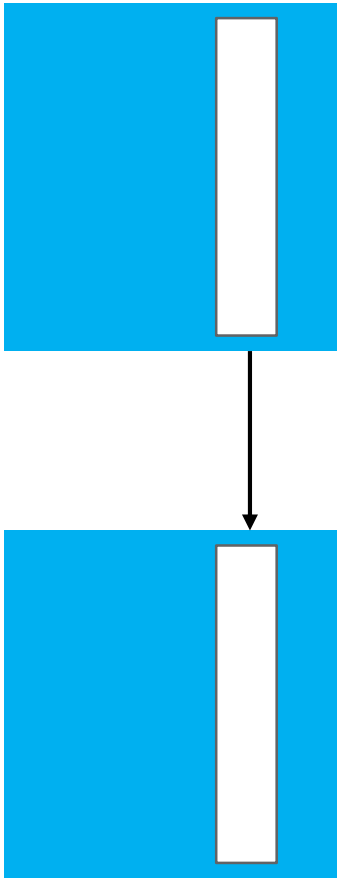
Sloupcové transformace v DataFrame

Sloupcové transformace

`DF.withColumn(`*`novy_sloupec, sloupcovy_vyraz`*`)`

sloupec DataFrame → sloupec DataFrame

něco jiného než map u RDD!



- základní aritmetika
  např.
  ```
  DF['col1'] = DF['col2'] * DF['col3']
  ```
- sloupcové transformační funkce
  nutno importovat Pyspark SQL functions
  např.
  ```
  from pyspark.sql import function as F
  DF['col1'] = F.lower(DF['col2'])
  ```

PROFINIT

Pyspark SQL functions – příklady

- **regexp_replace** (náhrada podle regulárního výrazu)
- **lower** (na malá písmena)
- **length** (délka řetězce)
- **split** (rozdělení řetězce – výsledek je array)
- **size** (počet prvků array – odpovídá funkci len)
- **when... otherwise** (sloupcové if-else)
- **lit** (konstanta)
- **udf** (uživatelská funkce – pokud nelze použít funkci Spark SQL)

Viz referenční příručka:
https://spark.apache.org/docs/2.4.0/api/python/pyspark.sql.html

PROFINIT

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