UC2

Stiahnuté a rozbalené datasety som vložil do zložky “work” (ako na cvičení), spustil Spark&Zeppelin cez docker (rovnaký postup ako na cvičeniach), a vytvoril nový notebook.

Predtým, ako som sa pustil do jednotlivých zadaní, som si dáta načítal:

```
// read
val ratings = spark.read.format("csv").option("sep", "\t").option("header", "true").option("inferSchema", "true").load("data.tsv")
val episode = spark.read.format("csv").option("sep", "\t").option("header", "true").option("inferSchema", "true").load("data-2.tsv")
val crew = spark.read.format("csv").option("sep", "\t").option("header", "true").option("inferSchema", "true").load("data-3.tsv")
val basics = spark.read.format("csv").option("sep", "\t").option("header", "true").option("inferSchema", "true").load("data-4.tsv")
val basics2 = spark.read.format("csv").option("sep", "\t").option("header", "true").option("inferSchema", "true").load("data-5.tsv")
val akas = spark.read.format("csv").option("sep", "\t").option("header", "true").option("inferSchema", "true").load("data-6.tsv")
val principals = spark.read.format("csv").option("sep", "\t").option("header", "true").option("inferSchema", "true").load("data-7.tsv")
```

SPARK JOBS FINISHED

```
ratings: org.apache.spark.sql.DataFrame = [tconst: string, averageRating: double ... 1 more field]
episode: org.apache.spark.sql.DataFrame = [tconst: string, parentTconst: string ... 2 more fields]
crew: org.apache.spark.sql.DataFrame = [tconst: string, directors: string ... 1 more field]
basics: org.apache.spark.sql.DataFrame = [tconst: string, titleType: string ... 7 more fields]
basics2: org.apache.spark.sql.DataFrame = [nconst: string, primaryName: string ... 4 more fields]
akas: org.apache.spark.sql.DataFrame = [titleId: string, ordering: int ... 6 more fields]
principals: org.apache.spark.sql.DataFrame = [tconst: string, ordering: int ... 4 more fields]
```

1)

```
// 1
ratings.show(10, false)
episode.show(10, false)
crew.show(10, false)
basics.show(10, false)
basics2.show(10, false)
akas.show(10, false)
principals.show(10, false)principals.show(10, false)
```

SPARK JOBS FINISHED

only showing top 10 rows

tconst	ordering	nconst	category	job	characters
tt0000001	1	nm1588970	self	null	["Self"]
tt0000001	2	nm0005690	director	null	null
tt0000001	3	nm0374658	cinematographer	director of photography	null
tt0000002	1	nm0721526	director	null	null
tt0000002	2	nm1335271	composer	null	null
tt0000003	1	nm0721526	director	null	null
tt0000003	2	nm5442194	producer	producer	null
tt0000003	3	nm1335271	composer	null	null
tt0000003	4	nm5442200	editor	null	null
tt0000004	1	nm0721526	director	null	null

only showing top 10 rows

Načítané dáta som jednoducho vypísal pomocou “premenná”.show(10, false) - false pre to, aby som videl všetko

2)

```
// 2
ratings.distinct.show(10, false)
episode.distinct.show(10, false)
crew.distinct.show(10, false)
basics.distinct.show(10, false)
basics2.distinct.show(10, false)
akas.distinct.show(10, false)
principals.distinct.show(10, false)
```

SPARK JOBS ABORT

tconst	averageRating	numVotes
tt0000357	5.8	346
tt0000662	4.8	18
tt0001044	5.7	80
tt0002266	6.3	6
tt0003565	5.8	21
tt0003860	5.1	76
tt0005190	7.5	11
tt0006101	7.6	9
tt0006333	6.2	1521
tt0006820	6.1	75

only showing top 10 rows

tconst	parentTconst	seasonNumber	episodeNumber

Rovnako ako v predošlom bode, na dataframy som použil .distinct() na zobrazenie jedinečných riadkov

3)

```
// 3
val types_count = akas.select("region", "language", "types").groupBy("region", "language", "types").count()
types_count.coalesce(1)
    .write.format("csv")
    .option("sep", ";")
    .option("header", "true")
    .save("types_count.csv")
```

SPARK JOB FINISHED

```
types_count: org.apache.spark.sql.DataFrame = [region: string, language: string ... 2 more fields]
```

```
root@6b62c4e00fd5:/work# head types_count.csv/part-00000-2d895284-c73c-41e2-b636
-2e66241b2d2e-c000.csv | cat
region;language;types;count
JP;en;working;83
TH;en;\N;110
FI;fi;\N;4
IL;ar;alternative;3
AE;\N;\N;1426
NG;\N;imdbDisplay;1
MD;\N;alternative;2
US;\N;alternative;16934
\N;\N;dvd;63
```

Na spočítanie počtu filmov odvíjajúceho sa od ich typu, regiónu a jazyka som spojil pomocou .groupBy stĺpce, ktoré boli vyžadované a .count() na ich spočítanie.
To som následne pomocou .write zapísal do súboru a jeho obsah som vypísal na konzole.
Coalesce(1) som využil z dôvodu, aby sa vytvoril len jeden súbor

4)

```
// 4
val czech_movies = akas.select("titleId", "title", "region").filter($"region" === "CZ")
val years = basics.select("tconst", "startYear", "originalTitle")
val cz_movies = czech_movies.join(years, years("tconst") === czech_movies("titleId"), "inner").sort
    ("startYear")

val crew_added = cz_movies.join(crew, Seq("tconst"), "inner")

val actors = principals.filter($"category" === "actor" || $"category" === "actress")
val countt = actors.groupBy("tconst").count()

val finallll = countt.join(crew_added, Seq("tconst"), "inner").orderBy(col("startYear")).drop("tconst",
    "titleId", "region", "startYear")
finallll.show(5)
```

SPARK JOBS FINISHED

```
+-----+--------------------+--------------------+------------------+--------------------+
|count|               title|       originalTitle|         directors|             writers|
+-----+--------------------+--------------------+------------------+--------------------+
|    2|    Pokropený kropic|   L'arroseur arrosé|nm0525910,nm0349785|                  \N|
|    1|    Dreyfusova aféra|   L'affaire Dreyfus|         nm0617588|           nm0617588|
|    4|      Cesta na Mesíc|Le voyage dans la...|         nm0617588|nm0617588,nm08945...|
|    7|Velká zeleznicní ...|The Great Train R...|         nm0692105| nm1145809,nm0692105|
|    4|    Cesta do nemozna|Le voyage à trave...|         nm0617588|nm0617588,nm08945...|
+-----+--------------------+--------------------+------------------+--------------------+
only showing top 5 rows
```

Na začiatok som si vyfiltroval české filmy (region === CZ) a spojil som ich s tabuľkou years (id, rok, názov filmu) podľa ID filmu, aby som mal tabuľku českých filmov s ich informáciami.

Ďalej som do tabuľky pridal režisérov a scénaristov (celá crew tabuľka) - join cez ID filmu, aby filmy, ktoré sú české, mali svoju crew.
Ako posledné som vyfiltroval z datasetu principals hercov (herec - category === actor/actress) a pomocou groupBy(id).count() som zistil, koľko hercov hralo v jednotlivých filmoch.
Teraz už len bolo treba spojiť tabuľku s počtami hercov s českými filmami, aby mali priradené počty a zoradiť od najstarších (zobrazených je len prvých 5, kvôli ušetreniu času som nechcel program spúšťať znovu s 10)

5)

```
// 5

val episodes = episode.join(basics, basics("tconst") === episode("tconst"), "left").drop(basics("tconst"))
    .filter($"startYear" >= 2005 && $"startYear" <= 2016 && $"runtimeMinutes"> 60)
    .select("tconst", "primaryTitle", "startYear", "runtimeMinutes")

val actors2 = principals.filter($"category" === "actor" || $"category" === "actress").select("nconst", "tconst")
val fqhwfwq = actors2.join(episodes, episodes("tconst") === actors2("tconst"), "inner").drop(actors2("tconst"))
val named_actors = fqhwfwq.join(basics2.select("nconst", "primaryName"), Seq("nconst"), "inner")
val qwhof = named_actors.groupBy("primaryTitle").agg(collect_list("primaryName").as("herci"))
qwhof.show(5, false)
```

SPARK JOB FINISHED

```
+------------------------------+-----------------------------------------------------------------------------------------+
|primaryTitle                  |herci                                                                                    |
+------------------------------+-----------------------------------------------------------------------------------------+
|Combat Jack                   |[Alexander Skarsgård, James Ransone, Jon Huertas, Lee Tergesen]                          |
|The Battle for Hitler's Supership|[Lincoln Fraser, Sarah Evetts, Andrea Ewing, Ian Carnegie, Helen Austin, Marcus Churchill]  |
|Dame de Carreau               |[Valérie Decobert-Koretzky, Jean-Toussaint Bernard, Sophie-Charlotte Husson, Thierry Godard]|
|Padnij                        |[Milena Lisiecka, Dorota Kolak, Marta Kalmus, Monika Chomicka]                           |
|Bienzle und der Sizilianer    |[Rita Russek, Rüdiger Wandel, Klaus Spürkel, Dietz Werner Steck]                         |
+------------------------------+-----------------------------------------------------------------------------------------+
only showing top 5 rows
```

Začal som tým, že som si spojil tabuľky episodes (obsahuje všetky epizódy a ich ID) a basics (obsahuje info, ktoré treba vypísať), aby som dostal tabuľku epizód, ktorá obsahuje ich názvy apod. Následne som si z nej vyfiltroval len tie epizódy, ktoré boli natočené v období od 2005 do 2016 a stopáž je viac ako 60 minút (príkaz .filter()) a vybral si z nej len dôležité stĺpce - ID, názov, rok a runtime (rok a runtime neskôr zahadzujem, no nechcel som do kódu už zasahovať, aby som na výsledok nečakal 15 minút)
Tabuľku hercov (actors2) som získal tak, že z datasetu principals som si vyfiltroval podľa stĺpca “category” tých, ktorí tam majú napísané “actor”/“actress” a vybral som z nej id herca a id filmu, v ktorom hral.
Následne som hercov spojil s tabuľkou “episodes”, čím mi vyšla tabuľka epizód a hercov, ktorí v nich hrali (ostatní sa nejoinli)
Keďže som chcel vypísať mená hercov, a nie len ich ID, tabuľku s epizódami a hercami som spojil s basics2 (dataset names.basics), čo mi dalo mená hercov.
A ako posledné som v tabuľke s priradenými menami hercov groupBy-op primaryTitle (názov epizódy) s menami hercov - v agregačnej funkcii collect_list, aby herci boli zapísaní v jednom liste pri filme.

6)

```
// 6
import org.apache.spark.sql.types.DoubleType

val top = ratings.select("tconst", "averageRating").filter($"averageRating" > 7.5)
val decade = basics.select("tconst", "startYear", "originalTitle").filter($"startYear" >= 1900 &&
    $"startYear" < 2000)

val czech_tmp = akas.select("titleId", "title", "region").filter($"region" === "CZ")
val czech_decade = czech_tmp.join(decade, decade("tconst") === czech_tmp("titleId"), "inner")
val cz_top = czech_decade.join(top, Seq("tconst"), "inner").drop("region", "startYear", "originalTitle",
    "tconst")
cz_top.show(5)
```

```
+---------+--------------------+-------------+
|  titleId|               title|averageRating|
+---------+--------------------+-------------+
|tt0032553|            Diktátor|          8.4|
|tt0032599|    Jeho dívka Pátek|          7.9|
|tt0120815|Zachraňte vojína ...|          8.6|
|tt0091251|      Jdi a dívej se|          8.3|
|tt0063522| Rosemary má děťátko|          8.0|
+---------+--------------------+-------------+
only showing top 5 rows
```

Začal som tým, že som si z datasetu hodnotení vybral tie hodnotenia, ktoré sú vyššie ako 7.5 (žiaľ neviem, čo je to kvartíl a nemal som čas to zisťovať, takže som to simuloval aspoň takto)
Z datasetu titles.basics som si vyfiltroval filmy natočené v 20. storočí
Tie som následne joinol s tabuľkou czech_tmp, ktorá obsahovala české filmy (region === CZ) a nakoniec som české filmy natočené v 20. storočí (czech_decade) joinol s top hodnotenými filmami (hodnotenie lepšie ako 7.5)

7)

—-

8)

```
// 8
val actors2 = principals.filter($"category" === "actor" || $"category" === "actress").select("nconst", "tconst")

val dead_people = basics2.filter($"deathYear" >= 1980 && $"deathYear" <= 2010).select("nconst")

val dead_actors = dead_people.join(actors2, actors2("nconst") === dead_people("nconst"), "left")

val dead_actors_final = dead_actors.drop("nconst").distinct()

val comedies = basics.filter($"genres".contains("Comedy"))
val eight_final = comedies.join(dead_actors_final, Seq("tconst"), "inner")

eight_final.show(5)
```

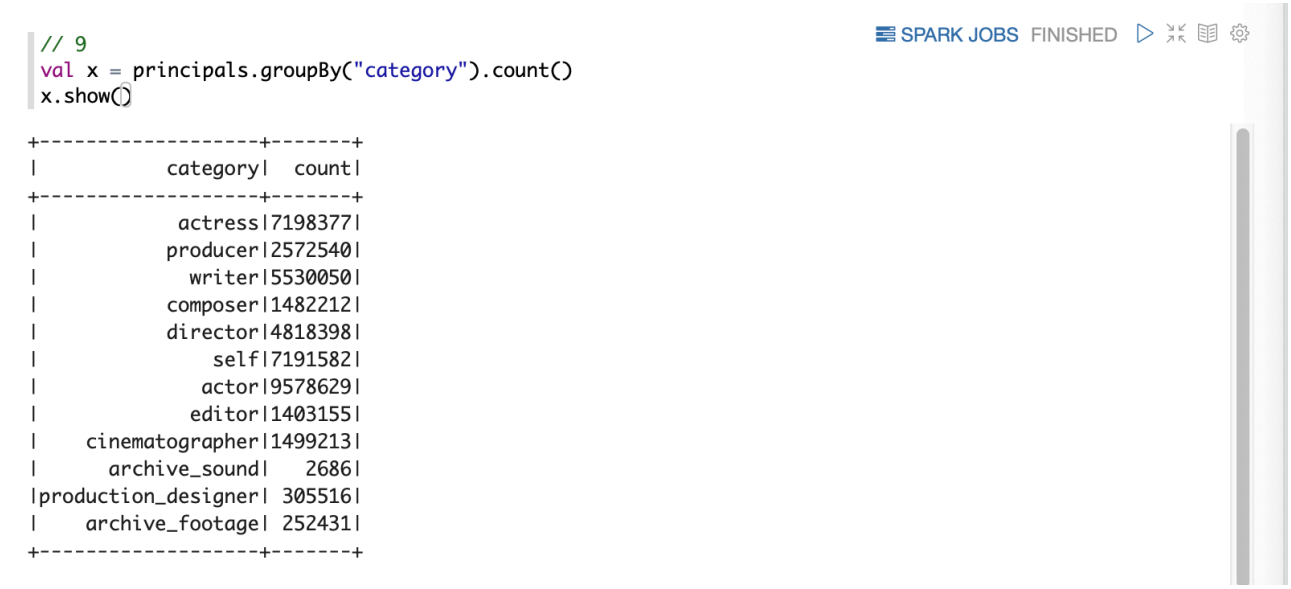
```
+---------+---------+--------------------+--------------------+-------+---------+-------+--------------+--------------------+
|   tconst|titleType|        primaryTitle|       originalTitle|isAdult|startYear|endYear|runtimeMinutes|              genres|
+---------+---------+--------------------+--------------------+-------+---------+-------+--------------+--------------------+
|tt0006441|    movie|  Blue Blood and Red|  Blue Blood and Red|      0|     1916|     \N|            50|      Comedy,Western|
|tt0010179|    movie|        God's Outlaw|        God's Outlaw|      0|     1919|     \N|            \N|Comedy,Drama,Western|
|tt0010752|    short|      Swat the Crook|      Swat the Crook|      0|     1919|     \N|            \N|         Comedy,Short|
|tt0011852|    movie|What's Your Husba...|What's Your Husba...|      0|     1920|     \N|            50|              Comedy|
|tt0012255|    short|   The Haunted House|   The Haunted House|      0|     1921|     \N|            21| Comedy,Horror,Short|
+---------+---------+--------------------+--------------------+-------+---------+-------+--------------+--------------------+
only showing top 5 rows

comedies: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [tconst: string, titleType: string ... 7 more fields]
eight_final: org.apache.spark.sql.DataFrame = [tconst: string, titleType: string ... 7 more fields]
```

Na zobrazenie filmov hercov, ktorí zomreli v rokoch 1980-2010 a filmy, v ktorých hrali, sú komédie, som si na začiatok vyfiltroval hercov (premenná actors2), rovnako ako v úlohe 5, a mŕtvych ľudí (dead_people) a tieto tabuľky spojil do jednej, aby som dostal mŕtvych hercov.
Nad tabuľkou mŕtvych hercov som ešte previedol .distinct(), nech sa filmy neopakujú viackrát.

Posledné, čo bolo treba, bolo vyfiltrovať komédie spomedzi všetkých filmov v datasete titles.basics (.filter($"genres".contains("Comedy"))) a spojiť túto tabuľku komédií s tabuľkou filmov, v ktorých hrali mŕtvy herci

9)



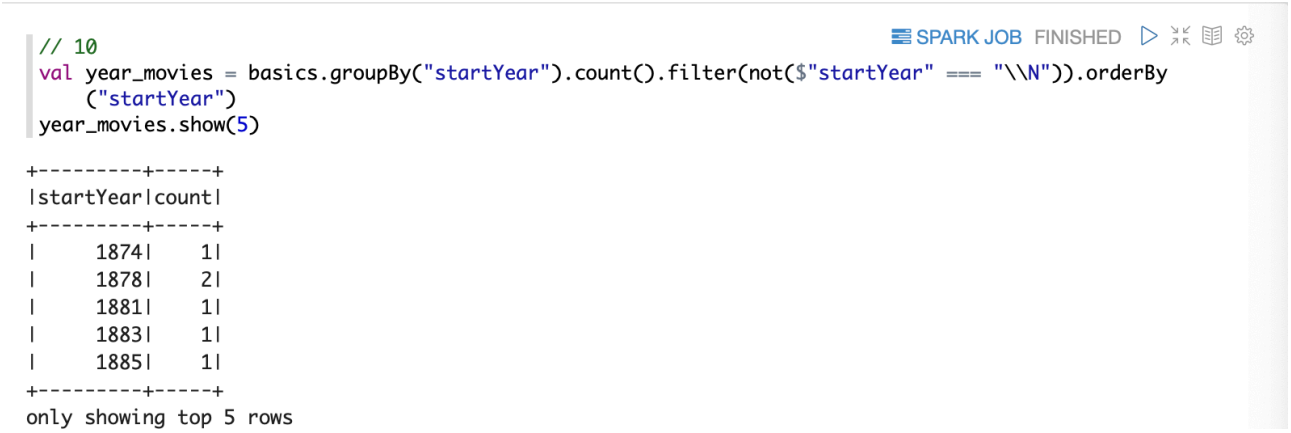
```
// 9
val x = principals.groupBy("category").count()
x.show()
```

SPARK JOBS FINISHED

```
+------------------+-------+
|          category|  count|
+------------------+-------+
|           actress|7198377|
|          producer|2572540|
|            writer|5530050|
|          composer|1482212|
|          director|4818398|
|              self|7191582|
|             actor|9578629|
|            editor|1403155|
|    cinematographer|1499213|
|      archive_sound|   2686|
|production_designer| 305516|
|    archive_footage| 252431|
+------------------+-------+
```

Zameral som sa na alternatívne zadanie bez potreby rozšíreného datasetu.. a pri tomto zadaní som len zgrupoval kategórie ľudí a spočítal počet ľudí, ktorí majú túto kategóriu zaradenú.

10)



```
// 10
val year_movies = basics.groupBy("startYear").count().filter(not($"startYear" === "\\N")).orderBy
    ("startYear")
year_movies.show(5)
```

SPARK JOB FINISHED

```
+---------+-----+
|startYear|count|
+---------+-----+
|     1874|    1|
|     1878|    2|
|     1881|    1|
|     1883|    1|
|     1885|    1|
+---------+-----+
only showing top 5 rows
```

Pred vytvorením vizualizácie som si vytvoril tabuľku obsahujúcu počet natočených filmov v danom roku. Filter (not($"startYear" === "\\N")) slúži na to, aby boli z tabuľky vyhodené NaN dáta.
Po vytvorení tabuľky som postupoval ako na cvičení - registerTempTable a cez %sql spustil command na vypísanie tabuľky (startYear ≥ 2010 podľa zadania)

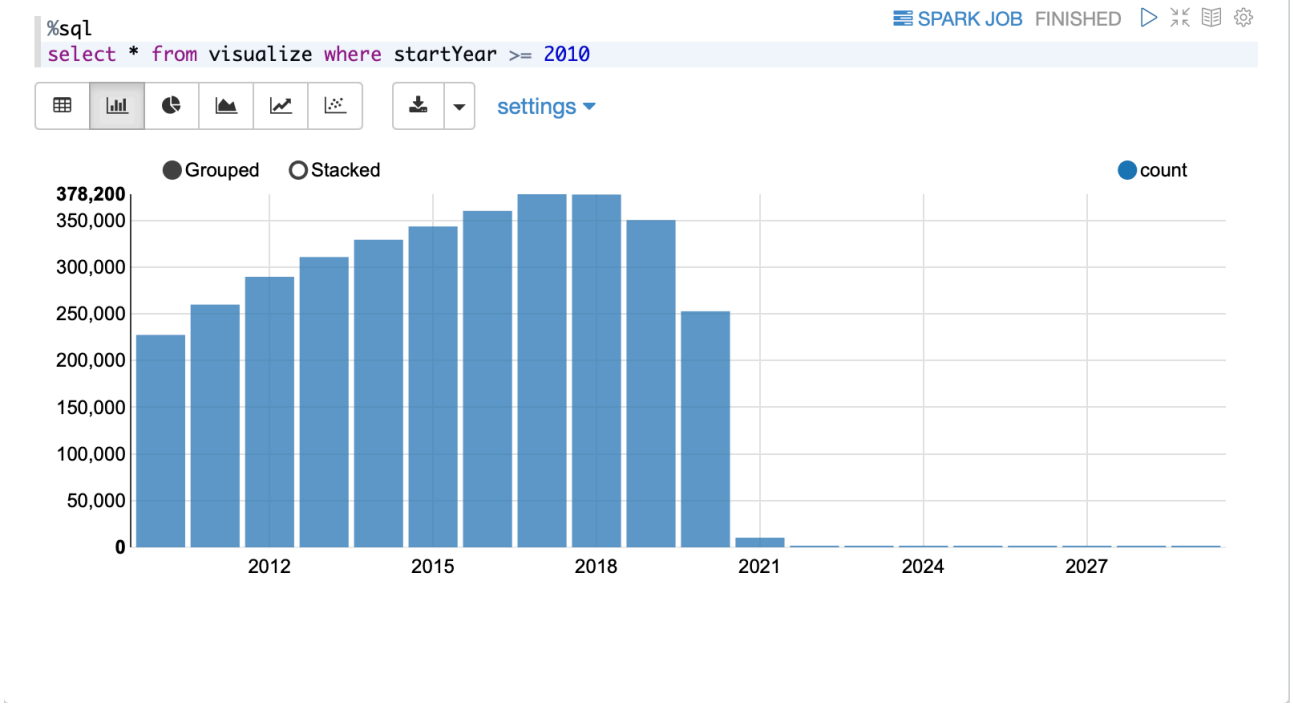
```
year_movies.registerTempTable("visualize")
```

FINISHED

```
warning: there was one deprecation warning; re-run with -deprecation for details
```

```
%sql
select * from visualize where startYear >= 2010
```

SPARK JOB FINISHED



Ospravedlňujem sa, že fotky neobsahujú môj watermark, ale toto je moje čestné prehlásenie, že screenshoty sú moje :)
A taktiež sa ospravedlňujem za prípadné posunuté obrázky/texty.. funkciu pridávania obrázkov do dokumentu musel vytvárať nejaký sadista, ktorý si užíva utrpenie ostatných.. a celý dokument som formátoval na 10 krát, nech to nejako vyzerá..