

Problem Set F Submission Form

Overview

Your Name	Hendi Kushta
Your SU Email	hkushta@syr.edu

Instructions

Put your name and SU email at the top. Answer these questions all from the lab. When asked to include screenshots, please follow the screen shot guidelines from the first homework.

Remember as you complete the homework it is not only about getting it right / correct. We will discuss the answers in class so it's important to articulate anything you would like to contribute to the discussion in your answer:

- If you feel the question is vague, include any assumptions you've made.
- If you feel the answer requires interpretation or justification provide it.
- If you do not know the answer to the question, articulate what you tried and how you are stuck.
- Highlight any doubts or questions you would like me to review.

This how you receive credit for answering questions which might not be correct. In addition, you must complete the reflection portion of the homework assignment for full credit. Since most answers will be similar this is an important part of your individual submission.

Complete Part II of this document first, then go back and complete the Reflection in Part I.

Part I - Reflection

Use this section to reflect on your learning. To achieve the highest grade on the assignment you must be as descriptive and personal as possible with your reflection.

1. As you completed this assignment, identify what you learned.

Use Apache Spark to import and export data into MongoDB. Perform CRUD operations with the MongoDB Query Language (MQL). Query MongoDB with Drill SQL, Spark SQL and MQL.

2. What barriers or challenges did you encounter while completing this assignment?

3. How prepared were you to complete this assignment? What can you do to be better prepared?

4. Rate your comfort level with this week's material. Use the rubric provided.

4 ==> I understand this material and can explain it to others.

3 ==> I understand this material.

2 ==> I somewhat understand the material but sometimes need guidance from others.

1 ==> I understand very little of this material and need extra help.

Part II – Questions

For each question, include a copy of the code required to complete the question along with a screenshot of the code and a screenshot of the output.

1. Use spark to load the **/datasets/json-samples/US-Senators.json** into the MongoDB database **labf** under the collection **senators**.

```
# Spark init
import pyspark
from pyspark.sql import SparkSession
mongo_uri = "mongodb://admin:mongopw@mongo:27017/admin?authSource=admin"

spark = SparkSession \
    .builder \
    .master("local") \
    .appName('jupyter-pyspark') \
    .config("spark.mongodb.input.uri", mongo_uri) \
    .config("spark.mongodb.output.uri", mongo_uri) \
    .config("spark.jars.packages", "org.mongodb.spark:mongo-spark-connector_2.12:3.0") \
    .getOrCreate()
sc = spark.sparkContext
sc.setLogLevel("ERROR")

df = spark.read.json("file:///home/jovyan/datasets/json-samples/US-Senators.json")
df.write.format("mongo") \
    .mode("overwrite") \
    .option("database", "labf") \
    .option("collection", "senators") \
    .save()
```

Mongo Express Database: labf

Viewing Database: labf

Collections

Collection Name + Create collection

View

Export

[JSON]

Import

senators

Del

Database Stats

Collections (incl. system.namespaces)	1
Data Size	117 KB
Storage Size	49.2 KB
Avg Obj Size #	1.18 KB
Objects #	99
Indexes #	1
Index Size	20.5 KB

2. From the Mongo Client, retrieve the firstname lastname , state and party for those senators in either the “Republican”, “Democrat, or “Independent” party.

```
labf> db.senators.find( {party:'Independent'}, { party:1, person: { firstname:2, lastname:3 }, state:4 })
[
  {
    _id: ObjectId('65d93b8b477fff3e5e3453a0'),
    party: 'Independent',
    person: { firstname: 'Bernard', lastname: 'Sanders' },
    state: 'VT'
  },
  {
    _id: ObjectId('65d93b8b477fff3e5e3453ae'),
    party: 'Independent',
    person: { firstname: 'Angus', lastname: 'King' },
    state: 'ME'
  }
]
```

3. From the Mongo Client, write an in MQL index to improve the query performance of question 2. Run the getIndexes() function on the collection to prove you created the index. Then use explain to prove the index is being used.

```
labf> db.senators.createIndex( {party:1} )
party_1
labf> db.senators.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { party: 1 }, name: 'party_1' }
]
```

```
stage: 'FETCH',
nReturned: 2,
executionTimeMillisEstimate: 0,
works: 3,
advanced: 2,
needTime: 0,
needYield: 0,
saveState: 0,
restoreState: 0,
isEOF: 1,
docsExamined: 2,
alreadyHasObj: 0,
inputStage: {
  stage: 'IXSCAN',
  nReturned: 2,
  executionTimeMillisEstimate: 0,
  works: 3,
  advanced: 2,
  needTime: 0,
  needYield: 0,
  saveState: 0,
  restoreState: 0,
  isEOF: 1,
  keyPattern: { party: 1 },
  indexName: 'party_1',
  isMultiKey: false,
  multiKeyPaths: { party: [] },
  isUnique: false,
  isSparse: false,
  isPartial: false,
  indexVersion: 2,
  direction: 'forward',
  indexBounds: { party: [ '['Independent', 'Independent'] ] },
  keysExamined: 2,
  seeks: 1,
  dupsTested: 0,
  dupsDropped: 0
}
```

4. Use Spark to Load in the **/datasets/netflix-canceled-2021/*.json** into Mongoddb database **labf** under the collection **nfcen**

```
nfcen = spark.read.option("multiline", True).json("file:///home/jovyan/datasets/netflix-canceled-2021/*.json")
nfcen.write.format("mongo") \
    .mode("overwrite") \
    .option("database", "labf") \
    .option("collection", "nfcen") \
    .save()
```

Mongo Express Database: labf

Viewing Database: labf

Collections

Collection Name + Create collection

View Export [JSON] Import nfcen

View Export [JSON] Import senators

Open *Un... Save

1 hkushta

Plain Text Tab Width: 8 Ln 1, Col 8 INS

5. Using Drill, write a query to list the name premier date and average rating of cancelled Netflix 2021 shows. Limit the shows to those with a average rating under 7.

Query type: ☒ SQL ☐ Physical ☐ Logical

Query

```
1 WITH table1 AS (
2     SELECT n.name, n.premiered, n.rating.average as rating
3     FROM mongo.labf.nfcen n
4 )
5 SELECT *
6 FROM table1
7 WHERE rating < 7;
8
```

Column visibility Show 10 entries Search:

name	premiered	rating
On My Block	2018-03-16	6.5
Mr. Iglesias	2019-06-21	6.8
The Crew	2021-02-15	6.2
Cursed	2020-07-17	6.2
Soocial	2019-04-12	6.7

6. Using Drill, write a query to list the name, premier date, and genre for only those shows in the History or Family genres. Make sure to include a column to display the name of the Genre.

Query type: ☒ SQL ☐ Physical ☐ Logical

Query

```
1 WITH table1 AS (
2     SELECT n.name, n.premiered, flatten(n.genres) AS genre
3     FROM mongo.labf.nfcan n
4 )
5
6 SELECT *
7 FROM table1
8 WHERE genre IN ('Family', 'History');
9
```

Column visibility	Show 10 entries	Search:
name	premiered	genre
Kim's Convenience	2016-10-11	Family
Peaky Blinders	2013-09-12	History
The Last Kingdom	2015-10-10	History
Country Comfort	2021-03-19	Family
Zero Chill	2021-03-15	Family

7. Using Spark or Spark SQL, create a DataFrame or view from the Netflix Cancellations MongoDB data, consisting of show name, season number, episode, number, episode name, airdate, and average rating (for the episode).

```
nfcan = spark.read.format("mongo").option("database", "labf").option("collection", "nfcan").load()

from pyspark.sql.functions import col, explode

tmp = nfcan.select( col("name").alias("showname"),
    explode("_embedded.episodes").alias("episode") )

episodes = tmp.select("showname",
    col("episode.name").alias("episode_name"),
    "episode.season",
    "episode.number",
    "episode.airdate",
    "episode.rating.average")

episodes.show(5)
```

showname	episode_name	season	number	airdate	average
Kim's Convenience	Gay Discount	1	1	2016-10-11	8.0
Kim's Convenience	Janet's Photos	1	2	2016-10-11	8.1
Kim's Convenience	Ddong Chim	1	3	2016-10-18	8.3
Kim's Convenience	Frank & Nayoung	1	4	2016-10-25	8.0
Kim's Convenience	Wingman	1	5	2016-11-01	8.1



only showing top 5 rows

8. Using the query you wrote in question 7 (if you want), write a Spark or Spark SQL query to get the lowest rated episodes of each season for the cancelled shows. Display show name, season number, episode number, episode name, and rating for that episode.
- NOTE: some shows have more than one episode with the lowest rating.

```
query = '''
with table1 as (
select
    showname,
    episode_name,
    season,
    number,
    airdate,
    average,
    min(average) over (partition by showname, season) as lowest Rated_season
from episodes
)

select *
from
    table1
where
    lowest Rated_season = average
order by
    showname,
    season
...

spark.sql(query).show(5)
```

Open ▾  *Un... Save  - □ ×

1 hkushta

Plain Text ▾ Tab Width: 8 ▾ Ln 1, Col 8 ▾ INS

```
[Stage 9:=====] (181 + 1) / 200
+-----+-----+-----+-----+-----+-----+-----+
|showname|episode_name|season|number|airdate|average|lowest Rated_season|
+-----+-----+-----+-----+-----+-----+-----+
|#blackAF|because of slavery|1|1|2020-04-17|5.3|5.3|
|#blackAF|because of slaver...|1|2|2020-04-17|5.3|5.3|
|Bonding|Penguins|1|6|2019-04-24|8.0|8.0|
|Bonding|Into the Woods|1|7|2019-04-24|8.0|8.0|
|Bonding|Old Friends, New ...|1|1|2019-04-24|8.0|8.0|
+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows
```

9. CHALLENGE YOURSELF! Display Name of show, a picture of the show, and show summary. Make it interactive so you can select the show and see the details.

```
from IPython.display import display, HTML, Image
from ipywidgets import interact, interact_manual


display(HTML("<h1>Netflix Cancelled Shows of 2021</h1>"))
shows = nfcان.select("name").distinct().sort("name").toPandas()["name"].values

@interact(show = shows)
def onchange(show):
    info = nfcان.select("name", "summary", "image.medium", "status", "rating.average").where(nfcان.name == show).toPandas().iloc[0]
    display(HTML(f"<h3>{info['name']}</h3>"))
    display(HTML(f"<p>STATUS: <b>{info['status']}</b> RATING: <b>{info['average']}</b>"))
    display(HTML(info['summary']))
    display(Image(url = info['medium']))
```

Netflix Cancelled Shows of 2021

show

Peaky Blinders
STATUS: **Running** RATING: **8.7**
An epic gangster drama set in the lawless streets of 1920s Birmingham.



Open Save
Plain Text Ln 1, Col 8