MidTerm Review

Anurag Nagar

CS 6307

Outline

1. Topics Covered
2. Introduction to Databases and SQL
3. Introduction to Big Data
4. NoSQL and MongoDB
   - NoSQL Concepts
   - MongoDB Concepts
5. MapReduce Concepts
   - Basics
   - PySpark Questions
   - Apache Spark
   - DataFrame Questions
6. Machine Learning

Topics Covered

List of topics covered so far:

- Introduction to databases and relational model
- Query processing and SQL
- NoSQL concepts and MongoDB
- Introduction to Big Data
- MapReduce using Scala
- Apache Spark, RDD, PairRDD, and DataFrame
- Machine Learning using Spark

Outline

1. Topics Covered
2. **Introduction to Databases and SQL**
3. Introduction to Big Data
4. NoSQL and MongoDB
   - NoSQL Concepts
   - MongoDB Concepts
5. MapReduce Concepts
   - Basics
   - PySpark Questions
   - Apache Spark
   - DataFrame Questions
6. Machine Learning

Introduction to Databases

Topics Covered

- Database - organized collection of data, grouped into tables

Introduction to Databases

Topics Covered

- Database - organized collection of data, grouped into tables
- Relational Database - tables are connected using relations and constraints

Introduction to Databases

Topics Covered

- Database - organized collection of data, grouped into tables
- Relational Database - tables are connected using relations and constraints
- Chief components of a relational database: **Entities** and **Relationships**

Introduction to Databases

Topics Covered

- Database - organized collection of data, grouped into tables
- Relational Database - tables are connected using relations and constraints
- Chief components of a relational database: **Entities** and **Relationships**
- Relational data model - model describing entities and their relationships

Structured Query Language (SQL)

Topics Covered

- Selection -> filtering rows (tuples or records)

Structured Query Language (SQL)

Topics Covered

- Selection -> filtering rows (tuples or records)
- Projection -> filtering columns (attributes)

Structured Query Language (SQL)

Topics Covered

- Selection -> filtering rows (tuples or records)
- Projection -> filtering columns (attributes)
- Join -> joining two tables on basis of common attributes

Structured Query Language (SQL)

Topics Covered

- Selection -> filtering rows (tuples or records)
- Projection -> filtering columns (attributes)
- Join -> joining two tables on basis of common attributes
- Group By and Aggregation -> generating aggregate statistics

Normalization

Topics Covered

- Understand 1st, 2nd, and 3rd normal forms

Normalization

Topics Covered

- Understand 1st, 2nd, and 3rd normal forms
- What do we achieve by normalization?

Normalization

Topics Covered

- Understand 1st, 2nd, and 3rd normal forms
- What do we achieve by normalization?
- What is the cost of normalization?

Questions

What is used to uniquely identify each record in a table?

1. Foreign Key
2. Primary Key
3. Field
4. Datatype

Questions

What is used to uniquely identify each record in a table?

1. Foreign Key
2. Primary Key
3. Field
4. Datatype

Questions

A relation (i.e. table) is in 1st NF if

1. Every attribute contains only atomic values
2. Every attribute contains only a numeric value
3. Every attribute contains any non-null value
4. Every attribute contains only a character value

Questions

A relation (i.e. table) is in 1st NF if

1. Every attribute contains only atomic values
2. Every attribute contains only a numeric value
3. Every attribute contains any non-null value
4. Every attribute contains only a character value

Questions

A relation (i.e. table) is in 2nd NF if

1. It is in the 1st NF
2. There is no attribute that doesn't depend on a part of the key
3. There is no attribute that doesn't depend on the entire key
4. There is no repeating value for an attribute

Questions

A relation (i.e. table) is in 2nd NF if

1. It is in the 1st NF
2. There is no attribute that doesn't depend on a part of the key
3. There is no attribute that doesn't depend on the entire key
4. There is no repeating value for an attribute

Questions

A relation (i.e. table) is in 2nd NF if

1. It is in the 1st NF
2. There is no attribute that doesn't depend on a part of the key
3. There is no attribute that doesn't depend on the entire key
4. There is no repeating value for an attribute

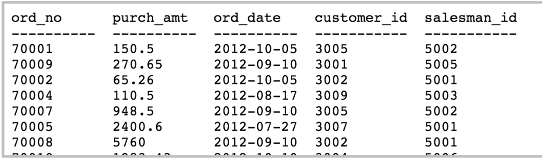
Questions

A relation (i.e. table) is in 2nd NF if

1. It is in the 1st NF
2. There is no attribute that doesn't depend on a part of the key
3. There is no attribute that doesn't depend on the entire key
4. There is no repeating value for an attribute

Questions

Consider the *orders* table below:

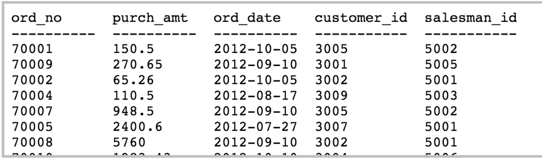


```
ord_no      purch_amt   ord_date    customer_id  salesman_id
----------  ----------  ----------  -----------  -----------
70001       150.5       2012-10-05  3005         5002
70009       270.65      2012-09-10  3001         5005
70002       65.26       2012-10-05  3002         5001
70004       110.5       2012-08-17  3009         5003
70007       948.5       2012-09-10  3005         5002
70005       2400.6      2012-07-27  3007         5001
70008       5760        2012-09-10  3002         5001
```

Write a query that will find the total **purch_amt** for each **customer_id**

Questions

Consider the *orders* table below:



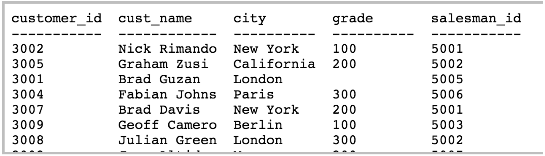
```
ord_no      purch_amt   ord_date    customer_id  salesman_id
----------  ----------  ----------  -----------  -----------
70001       150.5       2012-10-05  3005         5002
70009       270.65      2012-09-10  3001         5005
70002       65.26       2012-10-05  3002         5001
70004       110.5       2012-08-17  3009         5003
70007       948.5       2012-09-10  3005         5002
70005       2400.6      2012-07-27  3007         5001
70008       5760        2012-09-10  3002         5001
70010       1983.43     2012-10-10  3004         5006
```

Write a query that will find the total **purch_amt** for each **customer_id**

SELECT customer_id, SUM (purch_amt) FROM orders GROUP BY customer_id ;

Questions

Consider the *customer* table below:

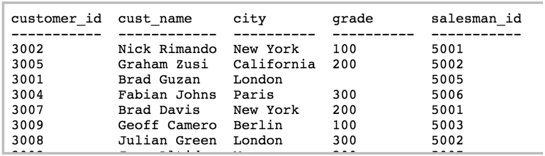


customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3005	Graham Zusi	California	200	5002
3001	Brad Guzan	London		5005
3004	Fabian Johns	Paris	300	5006
3007	Brad Davis	New York	200	5001
3009	Geoff Camero	Berlin	100	5003
3008	Julian Green	London	300	5002

Write a query that will find the highest **grade** for each **city**

Questions

Consider the *customer* table below:



```
customer_id  cust_name     city        grade       salesman_id
-----------  ------------  ----------  ----------  -----------
3002         Nick Rimando  New York    100         5001
3005         Graham Zusi   California  200         5002
3001         Brad Guzan    London                  5005
3004         Fabian Johns  Paris       300         5006
3007         Brad Davis    New York    200         5001
3009         Geoff Camero  Berlin      100         5003
3008         Julian Green  London      300         5002
```

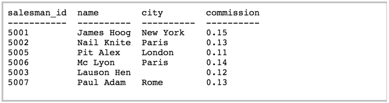
Write a query that will find the highest **grade** for each **city**

SELECT city, MAX(grade) FROM customer GROUP BY city;

Questions

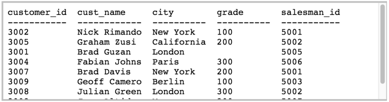
Consider two tables as shown below

Sample table: salesman



```
salesman_id  name        city        commission
-----------  ----------  ----------  ----------
5001         James Hoog  New York    0.15
5002         Nail Knite  Paris       0.13
5005         Pit Alex    London      0.11
5006         Mc Lyon     Paris       0.14
5003         Lauson Hen              0.12
5007         Paul Adam   Rome        0.13
```

Sample table: customer

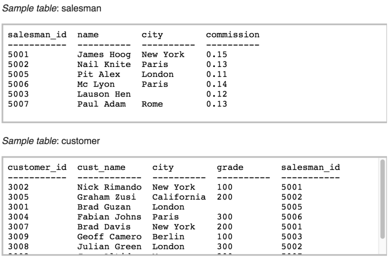


```
customer_id  cust_name     city        grade       salesman_id
-----------  ------------  ----------  ----------  -----------
3002         Nick Rimando  New York    100         5001
3005         Graham Zusi   California  200         5002
3001         Brad Guzan    London                  5005
3004         Fabian Johns  Paris       300         5006
3007         Brad Davis    New York    200         5001
3009         Geoff Camero  Berlin      100         5003
3008         Julian Green  London      300         5002
```

Write a query to return a list of customers and salesmen from the same city along with the city name

Questions

Consider two tables as shown below



Sample table: salesman

salesman_id	name	city	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5005	Pit Alex	London	0.11
5006	Mc Lyon	Paris	0.14
5003	Lauson Hen		0.12
5007	Paul Adam	Rome	0.13

Sample table: customer

customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3005	Graham Zusi	California	200	5002
3001	Brad Guzan	London		5005
3004	Fabian Johns	Paris	300	5006
3007	Brad Davis	New York	200	5001
3009	Geoff Camero	Berlin	100	5003
3008	Julian Green	London	300	5002

Write a query to return a list of customers and salesmen from the same city along with the city name

SELECT s.name, c.cust_name, s.city FROM salesman s INNER JOIN customer c ON s.city = c.city;

Outline

1. Topics Covered
2. Introduction to Databases and SQL
3. **Introduction to Big Data**
4. NoSQL and MongoDB
   - NoSQL Concepts
   - MongoDB Concepts
5. MapReduce Concepts
   - Basics
   - PySpark Questions
   - Apache Spark
   - DataFrame Questions
6. Machine Learning

Introduction to Big Data

What is Big Data?

- Remember the 3V definition

Introduction to Big Data

What is Big Data?

- Remember the 3V definition
- Examples of Big Data

Introduction to Big Data

What is Big Data?

- Remember the 3V definition
- Examples of Big Data
- Characteristics of Big Data e.g. raw data, log data, etc that needs to be processed to derive information

Introduction to Big Data

What is Big Data?

- Remember the 3V definition
- Examples of Big Data
- Characteristics of Big Data e.g. raw data, log data, etc that needs to be processed to derive information
- Go through the slides and reading assignment

Outline

1 Topics Covered

2 Introduction to Databases and SQL

3 Introduction to Big Data

4 NoSQL and MongoDB
- NoSQL Concepts
- MongoDB Concepts

5 MapReduce Concepts
- Basics
- PySpark Questions
- Apache Spark
- DataFrame Questions

6 Machine Learning

Questions

Which of the following are properties of a distributed system

1. Consistency
2. Availability
3. Partitionability
4. Duplication

Questions

Which of the following are properties of a distributed system

1. Consistency
2. Availability
3. Partitionability
4. Duplication

Questions

The CAP theorem states that:

For a distributed system, it is impossible to have more than 2 of the three CAP properties at the same time.

Questions

I have a movies collection with the following fields: genres, plot, runtime. I would like to run a query with following criteria:
Genres should be "Comedy"
The data should be sorted by runtime in descending order
I do not want to see the _id field.
Which query accomplishes this?

1. db.movies.find(genres: "Comedy",_id: 0, genres: 1, plot: 1, runtime: 1).sort(runtime:-1)
2. db.movies.find(genres: "Comedy").sort(runtime:-1)
3. db.movies.find(genres: "Comedy",_id: 0, genres: 1, plot: 1, runtime: 1).sort(runtime:1)
4. db.movies.find(genres: "Comedy", _id: 0, genres: 1, plot: 1, runtime: 1).sort(runtime:-1)

Questions

I have a movies collection with the following fields: genres, plot, runtime. I would like to run a query with following criteria:
Genres should be "Comedy"
The data should be sorted by runtime in descending order
I do not want to see the _id field.
Which query accomplishes this?

1. db.movies.find(genres: "Comedy",_id: 0, genres: 1, plot: 1, runtime: 1).sort(runtime:-1)
2. db.movies.find(genres: "Comedy").sort(runtime:-1)
3. db.movies.find(genres: "Comedy",_id: 0, genres: 1, plot: 1, runtime: 1).sort(runtime:1)
4. db.movies.find(genres: "Comedy", _id: 0, genres: 1, plot: 1, runtime: 1).sort(runtime:-1)

Practice many more such questions

Outline

1 Topics Covered

2 Introduction to Databases and SQL

3 Introduction to Big Data

4 NoSQL and MongoDB
- NoSQL Concepts
- MongoDB Concepts

5 MapReduce Concepts
- Basics
- PySpark Questions
- Apache Spark
- DataFrame Questions

6 Machine Learning

Transformations on Pair RDDs

Two phases:

- **Map** - Transformation from one list to another
- **Reduce** - Aggregates data

Questions

What is the output of the following code in Python?

```
odds = [3, 5, 7]
def myFun(x):
    return 2*x

result = map(lambda x: myFun(x) * x, odds)
print ( list ( result ))
```

Questions

What is the output of the following code in Python?

```
odds = [3, 5, 7]
def myFun(x):
    return 2*x

result = map(lambda x: myFun(x) * x, odds)
print ( list ( result ))
```

```
[18, 50, 98]
```

Questions

What is the output of the following code in Python?

```
odds = [3, 5, 7]
map(lambda x: x*x, odds)
print (odds)
```

Questions

What is the output of the following code in Python?

```
odds = [3, 5, 7]
map(lambda x: x*x, odds)
print(odds)
```

[3, 5, 7]

Questions

What will be the output of the following lines of code in PySpark:

```
num = sc.parallelize ([1, 2, 3])
num = map(lambda x: 2*x, num)
print (nums)
```

Questions

What will be the output of the following lines of code in PySpark:

```
num = sc.parallelize([1, 2, 3])
num = map(lambda x: 2*x, num)
print(nums)
```

It will produce an error. Think why?

Questions

Consider the Spark code snippet below:

```
storeAddress = sc. parallelize ([
["Ritual", "1026 Valencia St"], ["Philz", "748 Van Ness Ave"],
["Philz", "3101 24th St"], ["Starbucks", "Seattle"]])
```

Which of the following will return the count of each type of stores:

1. storeAddress.keys().distinct().count()
2. storeAddress.count()
3. storeAddress.keys().count()
4. storeAddress.map(lambda x: x[0]).distinct().count()

Questions

Consider the Spark code snippet below:

```
storeAddress = sc. parallelize ([
["Ritual", "1026 Valencia St"], ["Philz", "748 Van Ness Ave"],
["Philz", "3101 24th St"], ["Starbucks", "Seattle"]])
```

Which of the following will return the count of each type of stores:

1. storeAddress.keys().distinct().count()
2. storeAddress.count()
3. storeAddress.keys().count()
4. storeAddress.map(lambda x: x[0]).distinct().count()

Questions

Consider the Spark code snippet below.

```
storeAddress = sc. parallelize ([
["Ritual", "1026 Valencia St"], ["Philz", "748 Van Ness Ave"],
["Philz", "3101 24th St"], ["Starbucks", "Seattle"]])

storeRating = sc. parallelize ([["Ritual", 4.9], ["Philz", 4.8]])
```

How many elements will be there in the following:
storeAddress.join(storeRating)

1. 2
2. 3
3. 4
4. 0

Questions

Consider the Spark code snippet below.

```
storeAddress = sc. parallelize ([
["Ritual", "1026 Valencia St"], ["Philz", "748 Van Ness Ave"],
["Philz", "3101 24th St"], ["Starbucks", "Seattle"]])

storeRating = sc. parallelize ([["Ritual", 4.9], ["Philz", 4.8]])
```

How many elements will be there in the following:
storeAddress.join(storeRating)

1. 2
2. 3
3. 4
4. 0

Questions

Consider the Spark code snippet below.

```
storeRating = sc. parallelize ([
["Ritual", 4.9], ["Philz", 4.8], ["Philz", 4.0],
["Ritual", 2.5], ["Starbucks", 4.0]
]).toDF(['Store','Rating'])
```

You would like to find the **maximum** rating for all the stores. Which line accomplishes this?

1. storeRating.groupBy(’Store’).max(’Store’)
2. storeRating.max.reduceByKey()
3. storeRating.groupBy(’Store’).max(’Rating’)
4. storeRating.reduceByKey(lambda x, y : Math.max(x, y))

Questions

Consider the Spark code snippet below.

```
storeRating = sc.parallelize([
["Ritual", 4.9], ["Philz", 4.8], ["Philz", 4.0],
["Ritual", 2.5], ["Starbucks", 4.0]
]).toDF(['Store','Rating'])
```

You would like to find the **maximum** rating for all the stores. Which line accomplishes this?

1. storeRating.groupBy('Store').max('Store')
2. storeRating.max.reduceByKey()
3. storeRating.groupBy('Store').max('Rating')
4. storeRating.reduceByKey(lambda x, y : Math.max(x, y))

Questions

We would like to find the sum of elements of a list in Python. The first lines of code are given. Which of the choices finds the sum of elements?

```
from functools import reduce
 list = [2, 4, 8]
```

1. reduce(lambda x, y: x + y, list)
2. list.reduce(lambda x, y: x + y)
3. reduce(list, lambda x, y: x + y, list)
4. reduce(lambda x: x + y, list)

Questions

We would like to find the sum of elements of a list in Python. The first lines of code are given. Which of the choices finds the sum of elements?

```
from functools import reduce
 list = [2, 4, 8]
```

1. reduce(lambda x, y: x + y, list)
2. list.reduce(lambda x, y: x + y)
3. reduce(list, lambda x, y: x + y, list)
4. reduce(lambda x: x + y, list)

Apache Spark

Important features of Apache Spark project[1]:

- Open-source cluster computing framework

[1] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[1]:

- Open-source cluster computing framework
- Developed to provide real-time, low latency queries on data that is stored in a cluster, such as Hadoop

[1] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[1]:

- Open-source cluster computing framework
- Developed to provide real-time, low latency queries on data that is stored in a cluster, such as Hadoop
- Uses partitioned, and distributed in-memory datasets, known as Resilient Distributed Datasets (RDD) to speed up computation.

[1] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[1]:

- Open-source cluster computing framework
- Developed to provide real-time, low latency queries on data that is stored in a cluster, such as Hadoop
- Uses partitioned, and distributed in-memory datasets, known as Resilient Distributed Datasets (RDD) to speed up computation.
- Disk I/O, which is the limiting factor in case of traditional MapReduce algorithms, is avoided by using RDDs

[1] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[1]:

- Open-source cluster computing framework
- Developed to provide real-time, low latency queries on data that is stored in a cluster, such as Hadoop
- Uses partitioned, and distributed in-memory datasets, known as Resilient Distributed Datasets (RDD) to speed up computation.
- Disk I/O, which is the limiting factor in case of traditional MapReduce algorithms, is avoided by using RDDs
- Runs programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

[1] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[2]:

- Uses lazy evaluation for efficient processing

[2] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[2]:

- Uses lazy evaluation for efficient processing
- RDDs are immutable i.e. they cannot be updated once created

[2] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[2]:

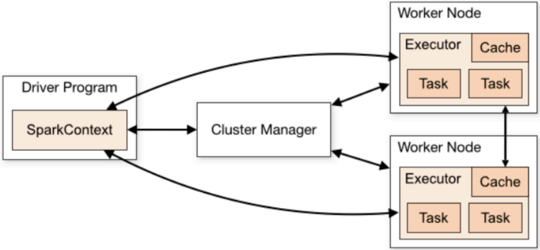
- Uses lazy evaluation for efficient processing
- RDDs are immutable i.e. they cannot be updated once created
- Spark core is the base engine for computation

[2] https://spark.apache.org/

Apache Spark

Important features of Apache Spark project[2]:

- Uses lazy evaluation for efficient processing
- RDDs are immutable i.e. they cannot be updated once created
- Spark core is the base engine for computation
- Spark workflow is shown below:



[2] https://spark.apache.org/

Questions

In Apache Spark, what is the use of the SparkContext (sc) object?

1. It represents a container for all the objects in memory
2. It represents all RDDs that are in your program
3. It represents an active connection to the Spark cluster and can be to request resources using the cluster manager
4. It represents the Hadoop file system

Questions

In Apache Spark, what is the use of the SparkContext (sc) object?

1. It represents a container for all the objects in memory
2. It represents all RDDs that are in your program
3. It represents an active connection to the Spark cluster and can be to request resources using the cluster manager
4. It represents the Hadoop file system

DataFrame Questions

Which of the following are true about DataFrames in Spark?[3]

1. They are part of the Spark SQL library
2. A DataFrame is a structured dataset organized into named columns
3. DataFrames can be constructued from a variety of sources, such as JSON files, CSV files, Hive tables or external databases
4. In Scala, a DataFrame is represented by a dataset of Rows

[3] See `https://spark.apache.org/docs/latest/sql-programming-guide.html#datasets-and-dataframes` for more details.

DataFrame Questions

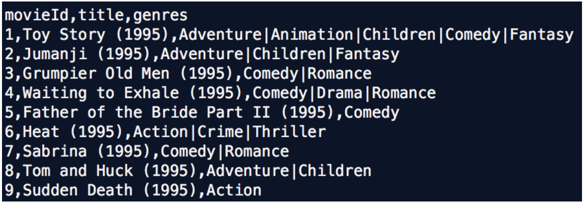
Which of the following are true about DataFrames in Spark?[3]

1. They are part of the Spark SQL library
2. A DataFrame is a structured dataset organized into named columns
3. DataFrames can be constructued from a variety of sources, such as JSON files, CSV files, Hive tables or external databases
4. In Scala, a DataFrame is represented by a dataset of Rows

[3] See `https://spark.apache.org/docs/latest/sql-programming-guide.html#datasets-and-dataframes` for more details.

DataFrame Questions

Suppose you have a file "movies.csv" :



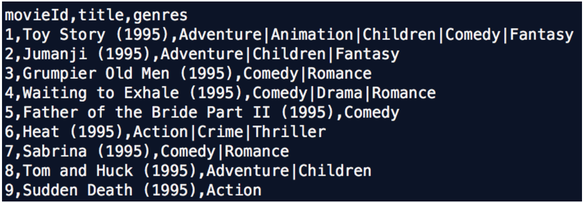
```
movieId,title,genres
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
4,Waiting to Exhale (1995),Comedy|Drama|Romance
5,Father of the Bride Part II (1995),Comedy
6,Heat (1995),Action|Crime|Thriller
7,Sabrina (1995),Comedy|Romance
8,Tom and Huck (1995),Adventure|Children
9,Sudden Death (1995),Action
```

Which of the following is the correct way to load this file into a DataFrame?

1. movies = spark.read.option("header","true").csv("movies.csv")
2. movies = spark.read.option("header","false").csv("movies.csv")
3. movies = spark.textFile.csv("movies.csv")
4. movies = spark.csv("movies.csv")

DataFrame Questions

Suppose you have a file "movies.csv" :



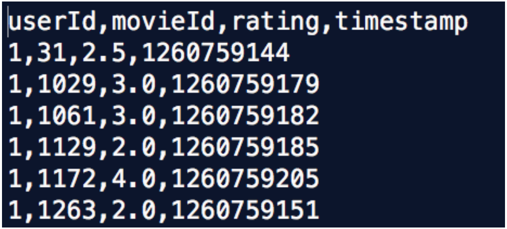
```
movieId,title,genres
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
4,Waiting to Exhale (1995),Comedy|Drama|Romance
5,Father of the Bride Part II (1995),Comedy
6,Heat (1995),Action|Crime|Thriller
7,Sabrina (1995),Comedy|Romance
8,Tom and Huck (1995),Adventure|Children
9,Sudden Death (1995),Action
```

Which of the following is the correct way to load this file into a DataFrame?

1. movies = spark.read.option("header","true").csv("movies.csv")
2. movies = spark.read.option("header","false").csv("movies.csv")
3. movies = spark.textFile.csv("movies.csv")
4. movies = spark.csv("movies.csv")

DataFrame Questions

Suppose you have a file "ratings.csv", which you have loaded into a **Dataframe** called **ratings**



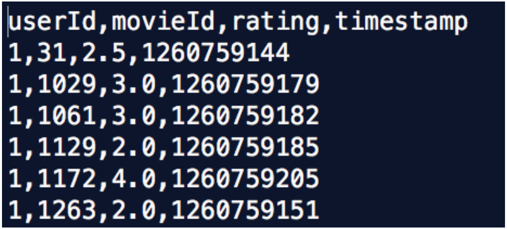
```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

How can you find out the number of ratings for each movieId?

1. ratings.reduceByKey("movieId").count()
2. ratings.groupBy("movieId").count()
3. ratings.groupBy("movieId").keys
4. ratings.groupBy("movieId").keys.count()

DataFrame Questions

Suppose you have a file "ratings.csv", which you have loaded into a **Dataframe** called **ratings**



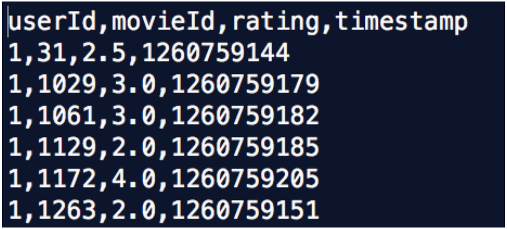
```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

How can you find out the number of ratings for each movieId?

1. ratings.reduceByKey("movieId").count()
2. ratings.groupBy("movieId").count()
3. ratings.groupBy("movieId").keys
4. ratings.groupBy("movieId").keys.count()

DataFrame Questions

Suppose you have a file "ratings.csv", which you have loaded into a **Dataframe** called **ratings**



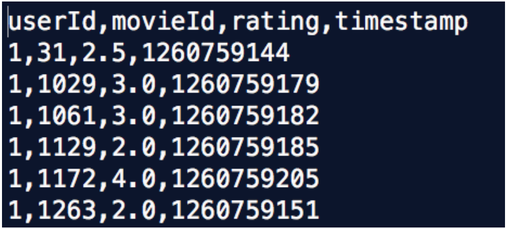
```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

You would like to find the **count** of ratings for each movieId sorted by descending order of count,

1. ratings.groupBy("movieId").agg(desc("count"))
2. ratings.groupBy("movieId").desc("count").show()
3. ratings.groupBy("movieId").count(). orderBy(desc("count"))
4. ratings.groupBy("movieId").orderBy(desc("count"))

DataFrame Questions

Suppose you have a file "ratings.csv", which you have loaded into a **Dataframe** called **ratings**



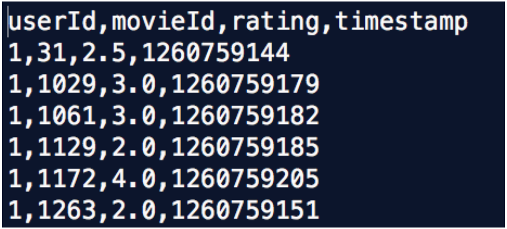
```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

You would like to find the **count** of ratings for each movieId sorted by descending order of count,

1. ratings.groupBy("movieId").agg(desc("count"))
2. ratings.groupBy("movieId").desc("count").show()
3. ratings.groupBy("movieId").count(). orderBy(desc("count"))
4. ratings.groupBy("movieId").orderBy(desc("count"))

DataFrame Questions

Suppose you have a file "ratings.csv", which you have loaded into a **Dataframe** called **ratings**



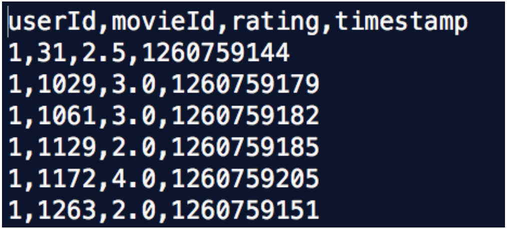
```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

You would like to find the **average** of ratings for each movieId sorted by descending order of average,

1. ratings.groupBy("movieId").avg("rating").sortBy(-1)
2. ratings.groupBy("movieId").agg(avg("rating").alias("avg")).orderBy(desc("avg"))
3. ratings.groupBy("movieId").avg("rating").orderBy(desc("avg"))
4. ratings.groupBy("movieId").avg("rating").orderDesc

DataFrame Questions

Suppose you have a file "ratings.csv", which you have loaded into a **Dataframe** called **ratings**



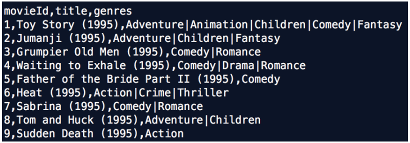
```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

You would like to find the **average** of ratings for each movieId sorted by descending order of average,

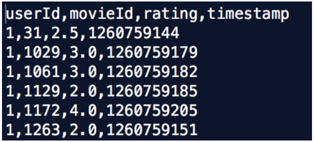
1. ratings.groupBy("movieId").avg("rating").sortBy(-1)
2. ratings.groupBy("movieId").agg(avg("rating").alias("avg")).orderBy(desc("avg"))
3. ratings.groupBy("movieId").avg("rating").orderBy(desc("avg"))
4. ratings.groupBy("movieId").avg("rating").orderDesc

DataFrame Questions

You have loaded the files below into DataFrames **movies** and **ratings**



```
movieId,title,genres
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
4,Waiting to Exhale (1995),Comedy|Drama|Romance
5,Father of the Bride Part II (1995),Comedy
6,Heat (1995),Action|Crime|Thriller
7,Sabrina (1995),Comedy|Romance
8,Tom and Huck (1995),Adventure|Children
9,Sudden Death (1995),Action
```



```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

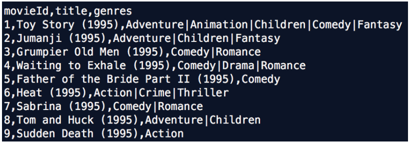
How would you join these two Dataframes? [4]

1. movies.join(ratings, movies.col("movieId") == ratings.col("movieId"))
2. movies.join(ratings, movies.col("movieId") === ratings.col("movieId"))
3. movies.join(ratings)
4. ratings.join(movies)

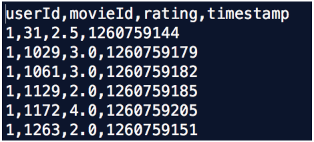
[4] See https://www.safaribooksonline.com/library/view/high-performance-spark/9781491943199/ch04.html for more details

DataFrame Questions

You have loaded the files below into DataFrames **movies** and **ratings**



```
movieId,title,genres
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
4,Waiting to Exhale (1995),Comedy|Drama|Romance
5,Father of the Bride Part II (1995),Comedy
6,Heat (1995),Action|Crime|Thriller
7,Sabrina (1995),Comedy|Romance
8,Tom and Huck (1995),Adventure|Children
9,Sudden Death (1995),Action
```



```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

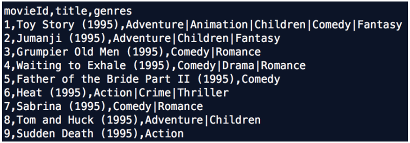
How would you join these two Dataframes? [4]

1. movies.join(ratings, movies.col("movieId") == ratings.col("movieId"))
2. movies.join(ratings, movies.col("movieId") === ratings.col("movieId"))
3. movies.join(ratings)
4. ratings.join(movies)

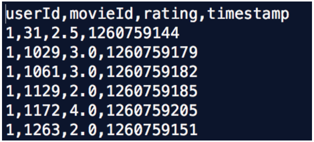
[4] See https://www.safaribooksonline.com/library/view/high-performance-spark/9781491943199/ch04.html for more details

DataFrame Questions

You have loaded the files below into DataFrames **movies** and **ratings**



```
movieId,title,genres
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
4,Waiting to Exhale (1995),Comedy|Drama|Romance
5,Father of the Bride Part II (1995),Comedy
6,Heat (1995),Action|Crime|Thriller
7,Sabrina (1995),Comedy|Romance
8,Tom and Huck (1995),Adventure|Children
9,Sudden Death (1995),Action
```



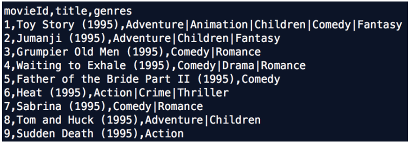
```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

You would like to find the **names** of the **top 5 highest rated movies**. Which of the following approaches would be **most efficient**?

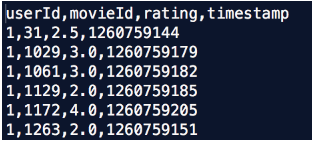
1. First join both Dataframes, compute avg for each movies, then sort by avg in descending order, and finally filter to top 5 rows.
2. First compute the avg for each movie, sort by avg in descending order and filter to top 5 rows, then join the filtered Dataframe to the movies DataFrame

DataFrame Questions

You have loaded the files below into DataFrames **movies** and **ratings**



```
movieId,title,genres
1,Toy Story (1995),Adventure|Animation|Children|Comedy|Fantasy
2,Jumanji (1995),Adventure|Children|Fantasy
3,Grumpier Old Men (1995),Comedy|Romance
4,Waiting to Exhale (1995),Comedy|Drama|Romance
5,Father of the Bride Part II (1995),Comedy
6,Heat (1995),Action|Crime|Thriller
7,Sabrina (1995),Comedy|Romance
8,Tom and Huck (1995),Adventure|Children
9,Sudden Death (1995),Action
```



```
userId,movieId,rating,timestamp
1,31,2.5,1260759144
1,1029,3.0,1260759179
1,1061,3.0,1260759182
1,1129,2.0,1260759185
1,1172,4.0,1260759205
1,1263,2.0,1260759151
```

You would like to find the **names** of the **top 5 highest rated movies**. Which of the following approaches would be **most efficient**?

1. First join both Dataframes, compute avg for each movies, then sort by avg in descending order, and finally filter to top 5 rows.
2. First compute the avg for each movie, sort by avg in descending order and filter to top 5 rows, then join the filtered Dataframe to the movies DataFrame

Outline

1 Topics Covered

2 Introduction to Databases and SQL

3 Introduction to Big Data

4 NoSQL and MongoDB
- NoSQL Concepts
- MongoDB Concepts

5 MapReduce Concepts
- Basics
- PySpark Questions
- Apache Spark
- DataFrame Questions

6 **Machine Learning**

Machine Learning

Which of the following are examples of Machine Learning?

1. Programming a home thermostat to start at a fixed time every day.
2. An application automatically learning to classify emails as personal, business, junk, or urgent
3. Creating an email rule that puts every email with "Lottery" in the subject to trash folder.
4. Obtaining movie suggestions from Netflix based on my viewing history
5. A machine that learns to classify clients as high, medium or low risk for default.

Machine Learning

Which of the following are examples of Machine Learning?

1. Programming a home thermostat to start at a fixed time every day.
2. An application automatically learning to classify emails as personal, business, junk, or urgent
3. Creating an email rule that puts every email with "Lottery" in the subject to trash folder.
4. Obtaining movie suggestions from Netflix based on my viewing history
5. A machine that learns to classify clients as high, medium or low risk for default.

Machine Learning

What are the three components of a ML system:

1. Experience (E), Task (T) and Performance measure (P)
2. Experience (E), Time (T) and Practice (P)
3. Work (W), ToDo (T) and Performance measure (P)
4. ELearning (E), Time (T) and Prediction (P)

Machine Learning

What are the three components of a ML system:

1. Experience (E), Task (T) and Performance measure (P)
2. Experience (E), Time (T) and Practice (P)
3. Work (W), ToDo (T) and Performance measure (P)
4. ELearning (E), Time (T) and Prediction (P)

Machine Learning

You are trying to train a machine to predict the amount of rainfall in mm based on weather conditions like humidity, temperature, etc. What type of machine learning is this?

1. Regression
2. Classification
3. Clustering
4. Recommender Systems

Machine Learning

You are trying to train a machine to predict the amount of rainfall in mm based on weather conditions like humidity, temperature, etc. What type of machine learning is this?

1. Regression
2. Classification
3. Clustering
4. Recommender Systems

Machine Learning

The library in Apache Spark that helps with Machine Learning is called ___________

1. MachineLibrary
2. MLlib
3. MAlib
4. MLlibraries

Machine Learning

The library in Apache Spark that helps with Machine Learning is called __________

1. MachineLibrary
2. MLlib
3. MAlib
4. MLlibraries

Machine Learning

Logistic Regression represents which type of Machine Learning

1. Regression
2. Classification
3. Recommender Systems
4. Clustering

Machine Learning

Logistic Regression represents which type of Machine Learning

1. Regression
2. Classification
3. Recommender Systems
4. Clustering

Machine Learning

Linear Regression represents which type of Machine Learning

1. Regression
2. Classification
3. Recommender Systems
4. Clustering

Machine Learning

Linear Regression represents which type of Machine Learning

1. Regression
2. Classification
3. Recommender Systems
4. Clustering

Questions

You would like to perform Logistic Regression on a dataset and use the code below:

```
val  train  = spark.read.csv("train.csv")
val  lr  = new LogisticRegression().setMaxIter(10)
~~l.setRegParam(0.3).setElasticNetParam(0.8)
```

Which of the following can be used to train the **lr** algorithm on the **train** dataset and obtain a trained model?

1. lr.train(train)
2. lr.fit(train)
3. lr.doTheTraining(train)
4. train.fit(lr)

Questions

You would like to perform Logistic Regression on a dataset and use the code below:

```
val train = spark.read.csv("train.csv")
val lr = new LogisticRegression().setMaxIter(10)
^^I.setRegParam(0.3).setElasticNetParam(0.8)
```

Which of the following can be used to train the **lr** algorithm on the **train** dataset and obtain a trained model?

1. lr.train(train)
2. lr.fit(train)
3. lr.doTheTraining(train)
4. train.fit(lr)

Questions

You would like to perform Logistic Regression on a dataset and use the code below:

```
val  train  = spark.read.csv("train.csv")
val  lr  = new LogisticRegression().setMaxIter(10)
~~l.setRegParam(0.3).setElasticNetParam(0.8)
# lr  is  trained  on the  train  dataset  to  obtain  model object
val  test  = spark.read("test.csv")
```

Which of the following can be used to test the lr model **model** on the **test** dataset?

1. model.transform(test)
2. model.fit(test)
3. model.doTheTesting(test)
4. test.fit(model)

Questions

You would like to perform Logistic Regression on a dataset and use the code below:

```
val train = spark.read.csv("train.csv")
val lr = new LogisticRegression().setMaxIter(10)
⌢⌢l.setRegParam(0.3).setElasticNetParam(0.8)
# lr is trained on the train dataset to obtain model object
val test = spark.read("test.csv")
```

Which of the following can be used to test the lr model **model** on the **test** dataset?

1. model.transform(test)
2. model.fit(test)
3. model.doTheTesting(test)
4. test.fit(model)