DATABRICKS-CERTIFIED-ASSOCIAT

Q&As

Databricks Certified Associate Developer for Apache Spark 3.0

Pass Databricks DATABRICKS-CERTIFIED-ASSOCIATE-DEVELOPER-FOR-APACHE-SPARK Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.pass2lead.com/databricks-certified-associate-developer-for-apache-spark.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Databricks Official Exam Center

QUESTION 1

Which of the following statements about DAGs is correct?

A. DAGs help direct how Spark executors process tasks, but are a limitation to the proper execution of a query when an executor fails.

B. DAG stands for "Directing Acyclic Graph".

C. Spark strategically hides DAGs from developers, since the high degree of automation in Spark means that developers never need to consider DAG layouts.

D. In contrast to transformations, DAGs are never lazily executed.

E. DAGs can be decomposed into tasks that are executed in parallel.

Correct Answer: E

QUESTION 2

In which order should the code blocks shown below be run in order to create a DataFrame that shows the mean of column predError of DataFrame transactionsDf per column storeId and productId, where productId should be either 2 or 3 and the returned DataFrame should be sorted in ascending order by column storeId, leaving out any nulls in that column?

DataFrame transactionsDf:

1.+-------------+---------+-----+-------+---------+----+

2.|transactionId|predError|value|storeId|productId| f|

3.+-------------+---------+-----+-------+---------+----+

4.| 1| 3| 4| 25| 1|null|

5.| 2| 6| 7| 2| 2|null|

6.| 3| 3| null| 25| 3|null|

7.| 4| null| null| 3| 2|null|

8.| 5| null| null| null| 2|null|

9.| 6| 3| 2| 25| 2|null|

10.+-------------+---------+-----+-------+---------+----+

1.

.mean("predError")

2.

.groupBy("storeId")

3.

.orderBy("storeId")

4.

transactionsDf.filter(transactionsDf.storeId.isNotNull())

5.

.pivot("productId", [2, 3])

A. 4, 5, 2, 3, 1

B. 4, 2, 1

C. 4, 1, 5, 2, 3

D. 4, 2, 5, 1, 3

E. 4, 3, 2, 5, 1

Correct Answer: D

Correct code block:

transactionsDf.filter(transactionsDf.storeId.isNotNull()).groupBy("storeId").pivot("productId", [2, 3]).mean

("predError").orderBy("storeId")

Output of correct code block:

+-------+----+----+

|storeId| 2| 3|

+-------+----+----+

| 2| 6.0|null|

| 3|null|null|

| 25| 3.0| 3.0|

+-------+----+----+

This is quite convoluted and requires you to think hard about the correct order of operations. The pivot

method also makes an appearance - a method that you may not know all that much about (yet).

At the first position in all answers is code block 4, so the is essentially just about the ordering of the

remaining 4 code blocks. The states that the returned DataFrame should be sorted by column storeId. So,

it should make sense to have code block 3 which includes the orderBy operator at the very end of the code

block. This leaves you with only two answer options. Now, it is useful to know more about the context of

pivot in PySpark. A common pattern is groupBy, pivot, and then another aggregating function, like mean.

In the documentation linked below you can see that pivot is a method of pyspark.sql.GroupedData meaning that before pivoting, you have to use groupBy. The only answer option matching this requirement

is the one in which code block 2 (which includes groupBy) is stated before code block 5 (which includes pivot).

More info: pyspark.sql.GroupedData.pivot -- PySpark 3.1.2 documentation

Static notebook | Dynamic notebook: See test 3, 43 (Databricks import instructions)

QUESTION 3

Which of the following code blocks creates a new 6-column DataFrame by appending the rows of the 6column DataFrame yesterdayTransactionsDf to the rows of the 6-column DataFrame todayTransactionsDf, ignoring that both DataFrames have different column names?

A. union(todayTransactionsDf, yesterdayTransactionsDf)

B. todayTransactionsDf.unionByName(yesterdayTransactionsDf, allowMissingColumns=True)

C. todayTransactionsDf.unionByName(yesterdayTransactionsDf)

D. todayTransactionsDf.concat(yesterdayTransactionsDf)

E. todayTransactionsDf.union(yesterdayTransactionsDf)

Correct Answer: E

todayTransactionsDf.union(yesterdayTransactionsDf) Correct. The union command appends rows of yesterdayTransactionsDf to the rows of todayTransactionsDf, ignoring that both DataFrames have different column names. The resulting DataFrame will have the column names of DataFrame todayTransactionsDf. todayTransactionsDf.unionByName (yesterdayTransactionsDf) No. unionByName specifically tries to match columns in the two DataFrames by name and only appends values in columns with identical names across the two DataFrames. In the form presented above, the command is a great fit for joining DataFrames that have exactly the same columns, but in a different order. In this case though, the command will fail because the two DataFrames have different columns. todayTransactionsDf.unionByName(yesterdayTransactionsDf, allowMissingColumns=True) No. The unionByName command is described in the previous explanation. However, with the allowMissingColumns argument set to True, it is no longer an issue that the two DataFrames have different column names. Any columns that do not have a match in the other DataFrame will be filled with null where there is no value. In the case at hand, the resulting DataFrame will have 7 or more columns though, so it this command is not the right answer. union(todayTransactionsDf, yesterdayTransactionsDf) No, there is no union method in pyspark.sql.functions. todayTransactionsDf.concat(yesterdayTransactionsDf) Wrong, the DataFrame class does not have a concat method. More info: pyspark.sql.DataFrame.union -- PySpark 3.1.2 documentation, pyspark.sql.DataFrame.unionByName -- PySpark 3.1.2 documentation Static notebook | Dynamic notebook: See test 3, 18 (Databricks import instructions)

QUESTION 4

Which of the following statements about Spark\\'s DataFrames is incorrect?

A. Spark\\'s DataFrames are immutable.

B. Spark\\'s DataFrames are equal to Python\\'s DataFrames.

C. Data in DataFrames is organized into named columns.

D. RDDs are at the core of DataFrames.

E. The data in DataFrames may be split into multiple chunks.

Correct Answer: B

Spark\\'s DataFrames are equal to Python\\'s or R\\'s DataFrames. No, they are not equal. They are only similar. A major difference between Spark and Python is that Spark\\'s DataFrames are distributed, whereby Python\\'s are not.

QUESTION 5

The code block shown below should return a new 2-column DataFrame that shows one attribute from column attributes per row next to the associated itemName, for all suppliers in column supplier whose name includes Sports. Choose the answer that correctly fills the blanks in the code block to accomplish this.

Sample of DataFrame itemsDf:

1.+------+----------------------------------+---------------------------+------------------+

2.|itemId|itemName |attributes |supplier |

3.+------+----------------------------------+---------------------------+------------------+

4.|1 |Thick Coat for Walking in the Snow|[blue, winter, cozy] |Sports Company Inc.| 5.|2 |Elegant Outdoors Summer Dress |[red, summer, fresh, cooling]|YetiX |

6.|3 |Outdoors Backpack |[green, summer, travel] |Sports Company Inc.|

7.+------+----------------------------------+---------------------------+------------------+

Code block:

itemsDf.__1__(__2__).select(__3__, __4__) A. 1. filter

2.

col("supplier").isin("Sports")

3.

"itemName"

4.

explode(col("attributes"))

B. 1. where

2.

col("supplier").contains("Sports")

3.

"itemName"

4.

"attributes"

C. 1. where

2.

col(supplier).contains("Sports")

3.

explode(attributes)

4.

itemName

D. 1. where

2.

"Sports".isin(col("Supplier"))

3.

"itemName"

4.

array_explode("attributes")

E. 1. filter

2.

col("supplier").contains("Sports")

3.

"itemName"

4.

explode("attributes")

Correct Answer: E

Output of correct code block:

+----------+------+

|itemName |col |

+----------+------+

|Thick Coat for Walking in the Snow|blue |

|Thick Coat for Walking in the Snow|winter|

|Thick Coat for Walking in the Snow|cozy |

|Outdoors Backpack |green |

|Outdoors Backpack |summer|

|Outdoors Backpack |travel|

+----------+------+

The key to solving this is knowing about Spark\\'s explode operator. Using this operator, you can extract values from arrays into single rows. The following guidance steps through the answers systematically from the first to the last gap. Note that there are many ways to solving the gap

---

QUESTION 6

The code block shown below should add a column itemNameBetweenSeparators to DataFrame itemsDf. The column should contain arrays of maximum 4 strings. The arrays should be composed of the values in column itemsDf which are separated at - or whitespace characters. Choose the answer that correctly fills the blanks in the code block to accomplish this.

Sample of DataFrame itemsDf:

1.+------+----------+------+

2.|itemId|itemName |supplier |

3.+------+----------+------+

4.|1 |Thick Coat for Walking in the Snow|Sports Company Inc.|

5.|2 |Elegant Outdoors Summer Dress |YetiX |

6.|3 |Outdoors Backpack |Sports Company Inc.|

7.+------+----------+------+

Code block:

itemsDf.__1__(__2__, __3__(__4__, "[\s\-]", __5__))

A. 1. withColumn

2.

"itemNameBetweenSeparators"

3.

split

4.

"itemName"

5.

4

(Correct)

B. 1. withColumnRenamed

2.

"itemNameBetweenSeparators"

3.

split

4.

"itemName"

5.

4

C. 1. withColumnRenamed

2.

"itemName"

3.

split

4.

"itemNameBetweenSeparators"

5.

4

D. 1. withColumn

2.

"itemNameBetweenSeparators"

3.

split

4.

"itemName"

5.

5

E. 1. withColumn

2.

itemNameBetweenSeparators

3.

str_split

4.

"itemName"

5.

5

Correct Answer: A

QUESTION 7

Which of the following code blocks uses a schema fileSchema to read a parquet file at location filePath into a DataFrame?

A. spark.read.schema(fileSchema).format("parquet").load(filePath)

B. spark.read.schema("fileSchema").format("parquet").load(filePath)

C. spark.read().schema(fileSchema).parquet(filePath)

D. spark.read().schema(fileSchema).format(parquet).load(filePath)

E. spark.read.schema(fileSchema).open(filePath)

Correct Answer: A

Pay attention here to which variables are quoted. fileSchema is a variable and thus should not be in quotes. parquet is not a variable and therefore should be in quotes. SparkSession.read (here referenced as spark.read) returns a DataFrameReader which all subsequent calls reference - the DataFrameReader is not callable, so you should not use parentheses here. Finally, there is no open method in PySpark. The method name is load. Static notebook | Dynamic notebook: See test 1, 44 (Databricks import instructions)

QUESTION 8

Which of the following code blocks returns a DataFrame that matches the multi-column DataFrame itemsDf, except that integer column itemId has been converted into a string column?

A. itemsDf.withColumn("itemId", convert("itemId", "string"))

B. itemsDf.withColumn("itemId", col("itemId").cast("string"))

C. itemsDf.select(cast("itemId", "string"))

D. itemsDf.withColumn("itemId", col("itemId").convert("string"))

E. spark.cast(itemsDf, "itemId", "string")

Correct Answer: B

itemsDf.withColumn("itemId", col("itemId").cast("string")) Correct. You can convert the data type of a column using the cast method of the Column class. Also note that you will have to use the withColumn method on itemsDf for replacing the existing itemId column with the new version that contains strings. itemsDf.withColumn("itemId", col("itemId").convert ("string")) Incorrect. The Column object that col("itemId") returns does not have a convert method. itemsDf.withColumn("itemId", convert("itemId", "string")) Wrong. Spark\\'s spark.sql.functions module does not have a convert method. The is trying to mislead you by using the word "converted". Type conversion is also called "type casting". This may help you remember to look for a cast method instead of a convert method (see correct answer). itemsDf.select(astype("itemId", "string")) False. While astype is a method of Column (and an alias of Column.cast), it is not a method of pyspark.sql.functions (what the code block implies). In addition, the

---

QUESTION 9

Which of the following code blocks reorders the values inside the arrays in column attributes of DataFrame

itemsDf from last to first one in the alphabet?

1.+------+-----------------------------+-------------------+

2.|itemId|attributes |supplier |

3.+------+-----------------------------+-------------------+

4.|1 |[blue, winter, cozy] |Sports Company Inc.|

5.|2 |[red, summer, fresh, cooling]|YetiX |

6.|3 |[green, summer, travel] |Sports Company Inc.|

7.+------+----------------------------+------------------+

A. itemsDf.withColumn(\\'attributes\\', sort_array(col(\\'attributes\\').desc()))

B. itemsDf.withColumn(\\'attributes\\', sort_array(desc(\\'attributes\\')))

C. itemsDf.withColumn(\\'attributes\\', sort(col(\\'attributes\\'), asc=False))

D. itemsDf.withColumn("attributes", sort_array("attributes", asc=False))

E. itemsDf.select(sort_array("attributes"))

Correct Answer: D

QUESTION 10

Which of the following code blocks returns a new DataFrame with the same columns as DataFrame transactionsDf, except for columns predError and value which should be removed?

A. transactionsDf.drop(["predError", "value"])

B. transactionsDf.drop("predError", "value")

C. transactionsDf.drop(col("predError"), col("value"))

D. transactionsDf.drop(predError, value)

E. transactionsDf.drop("predError and value")

Correct Answer: B

QUESTION 11

Which of the following code blocks efficiently converts DataFrame transactionsDf from 12 into 24 partitions?

A. transactionsDf.repartition(24, boost=True)

B. transactionsDf.repartition()

C. transactionsDf.repartition("itemId", 24)

D. transactionsDf.coalesce(24)

E. transactionsDf.repartition(24)

Correct Answer: E

QUESTION 12

Which of the elements in the labeled panels represent the operation performed for broadcast variables?

Larger image

A. 2, 5

B. 3

C. 2, 3

D. 1, 2

E. 1, 3, 4

Correct Answer: C

2,3 Correct! Both panels 2 and 3 represent the operation performed for broadcast variables. While a broadcast operation may look like panel 3, with the driver being the bottleneck, it most probably looks like panel 2. This is because the torrent protocol sits behind Spark\\'s broadcast implementation. In the torrent protocol, each executor will try to fetch missing broadcast variables from the driver or other nodes, preventing the driver from being the bottleneck. 1,2 Wrong. While panel 2 may represent broadcasting, panel 1 shows bi-directional communication which does not occur in broadcast operations. No. While broadcasting may materialize like shown in panel 3, its use of the torrent protocol also enables communciation as shown in panel 2 (see first explanation). 1,3,4 No. While panel 2 shows broadcasting, panel 1 shows bi-directional communication ?not a characteristic of broadcasting. Panel 4 shows uni-directional communication, but in the wrong direction. Panel 4 resembles more an accumulator variable than a broadcast variable. 2,5 Incorrect. While panel 2 shows broadcasting, panel 5 includes bi-directional communication ?not a characteristic of broadcasting. More info: Broadcast Join with Spark ?henning.kropponline.de

QUESTION 13

Which of the following code blocks performs an inner join between DataFrame itemsDf and DataFrame transactionsDf, using columns itemId and transactionId as join keys, respectively?

A. itemsDf.join(transactionsDf, "inner", itemsDf.itemId == transactionsDf.transactionId)

B. itemsDf.join(transactionsDf, itemId == transactionId)

C. itemsDf.join(transactionsDf, itemsDf.itemId == transactionsDf.transactionId, "inner")

D. itemsDf.join(transactionsDf, "itemsDf.itemId == transactionsDf.transactionId", "inner")

E. itemsDf.join(transactionsDf, col(itemsDf.itemId) == col(transactionsDf.transactionId))

Correct Answer: C

More info: pyspark.sql.DataFrame.join -- PySpark 3.1.2 documentation Static notebook | Dynamic notebook: See test 2, 27 (Databricks import instructions)

QUESTION 14

Which of the following statements about Spark\\'s configuration properties is incorrect?

A. The maximum number of tasks that an executor can process at the same time is controlled by the spark.task.cpus property.

B. The maximum number of tasks that an executor can process at the same time is controlled by the spark.executor.cores property.

C. The default value for spark.sql.autoBroadcastJoinThreshold is 10MB.

D. The default number of partitions to use when shuffling data for joins or aggregations is 300.

E. The default number of partitions returned from certain transformations can be controlled by the spark.default.parallelism property.

Correct Answer: D

QUESTION 15

The code block displayed below contains an error. The code block below is intended to add a column itemNameElements to DataFrame itemsDf that includes an array of all words in column itemName. Find the error.

Sample of DataFrame itemsDf:

1.+------+----------------------------------+-------------------+

2.|itemId|itemName |supplier |

3.+------+----------------------------------+-------------------+ 4.|1 |Thick Coat for Walking in the Snow|Sports

Company Inc.|

5.|2 |Elegant Outdoors Summer Dress |YetiX |

6.|3 |Outdoors Backpack |Sports Company Inc.|

7.+------+----------------------------------+-------------------+

Code block:

itemsDf.withColumnRenamed("itemNameElements", split("itemName"))

A. All column names need to be wrapped in the col() operator.

B. Operator withColumnRenamed needs to be replaced with operator withColumn and a second argument "," needs to be passed to the split method.

C. Operator withColumnRenamed needs to be replaced with operator withColumn and the split method needs to be replaced by the splitString method.

D. Operator withColumnRenamed needs to be replaced with operator withColumn and a second argument " " needs to be passed to the split method. E. The expressions "itemNameElements" and split("itemName") need to be swapped.

Correct Answer: D