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.geekcert.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

The code block shown below should return all rows of DataFrame itemsDf that have at least 3 items in column itemNameElements. Choose the answer that correctly fills the blanks in the code block to accomplish this.

Example of DataFrame itemsDf:

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

2.|itemId|itemName |supplier |itemNameElements |

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

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

5.|2 |Elegant Outdoors Summer Dress |YetiX |[Elegant, Outdoors, Summer, Dress] |

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

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

Code block:

itemsDf.__1__(__2__(__3__)__4__)

A. 1. select

2.

count

3.

col("itemNameElements")

4.

>3

B. 1. filter

2.

count

3.

itemNameElements

4.

>=3

C. 1. select

2.

count

3.

"itemNameElements"

4.

>3

D. 1. filter

2.

size

3.

"itemNameElements"

4.

>=3

E. 1. select

2.

size

3.

"itemNameElements"

4.

>3

Correct Answer: D

Correct code block:

itemsDf.filter(size("itemNameElements")>3)

Output of code block:

+------+---------------------------------+-----------------+---------------------------------------+ |itemId|itemName |

supplier |itemNameElements |

+------+---------------------------------+-----------------+---------------------------------------+ |1 |Thick Coat for

Walking in the Snow|Sports Company Inc.|[Thick, Coat, for, Walking, in, the, Snow]|

|2 |Elegant Outdoors Summer Dress |YetiX |[Elegant, Outdoors, Summer, Dress] |

+-----+---------------------------------+------------------+--+ The big difficulty with this is

in knowing the difference between count and size (refer to documentation below). size is the correct

function to choose here since it returns the number

of elements in an array on a per-row basis.

The other consideration for solving this is the difference between select and filter. Since we want to return

the rows in the original DataFrame, filter is the right choice. If we would use select, we would simply get a

single-column DataFrame showing which rows match the criteria, like so:

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

|(size(itemNameElements) > 3)|

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

|true |

|true |

|false |

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

More info:

Count documentation: pyspark.sql.functions.count -- PySpark 3.1.1 documentation Size documentation:

pyspark.sql.functions.size -- PySpark 3.1.1 documentation Static notebook | Dynamic notebook: See test

1, 47 (Databricks import instructions)

QUESTION 2

Which of the following code blocks returns a copy of DataFrame itemsDf where the column supplier has been renamed to manufacturer?

A. itemsDf.withColumn(["supplier", "manufacturer"])

B. itemsDf.withColumn("supplier").alias("manufacturer")

C. itemsDf.withColumnRenamed("supplier", "manufacturer")

D. itemsDf.withColumnRenamed(col("manufacturer"), col("supplier"))

E. itemsDf.withColumnsRenamed("supplier", "manufacturer")

Correct Answer: C

itemsDf.withColumnRenamed("supplier", "manufacturer") Correct! This uses the relatively trivial

DataFrame method withColumnRenamed for renaming column supplier to column manufacturer.

Note that the asks for "a copy of DataFrame itemsDf". This may be confusing if you are not familiar with

Spark yet. RDDs (Resilient Distributed Datasets) are the foundation of

Spark DataFrames and are immutable. As such, DataFrames are immutable, too. Any command that

changes anything in the DataFrame therefore necessarily returns a copy, or a new version, of it

that has the changes applied.

itemsDf.withColumnsRenamed("supplier", "manufacturer") Incorrect. Spark\\'s DataFrame API does not

have a withColumnsRenamed() method. itemsDf.withColumnRenamed(col("manufacturer"), col

("supplier")) No. Watch out ?although the col() method works for many methods of the DataFrame API,

withColumnRenamed is not one of them. As outlined in the documentation linked below,

withColumnRenamed expects strings.

itemsDf.withColumn(["supplier", "manufacturer"])

Wrong. While DataFrame.withColumn() exists in Spark, it has a different purpose than renaming columns.

withColumn is typically used to add columns to DataFrames, taking the name of the new

column as a first, and a Column as a second argument. Learn more via the documentation that is linked

below.

itemsDf.withColumn("supplier").alias("manufacturer") No. While DataFrame.withColumn() exists, it

requires 2 arguments. Furthermore, the alias() method on DataFrames would not help the cause of

renaming a column much.

DataFrame.alias() can be

useful in addressing the input of join statements. However, this is far outside of the scope of this question.

If you are curious nevertheless, check out the link below. More info:

pyspark.sql.DataFrame.withColumnRenamed -- PySpark 3.1.1 documentation,

pyspark.sql.DataFrame.withColumn -- PySpark 3.1.1 documentation, and pyspark.sql.DataFrame.alias --

PySpark 3.1.2 documentation (https://bit.ly/3aSB5tm , https://bit.ly/2Tv4rbE , https://bit.ly/2RbhBd2)

Static notebook | Dynamic notebook: See test 1, 31 (Databricks import instructions) (https://flrs.github.io/

spark_practice_tests_code/#1/31.html , https://bit.ly/sparkpracticeexams_import_instructions)

QUESTION 3

Which of the following describes the role of the cluster manager?

A. The cluster manager schedules tasks on the cluster in client mode.

B. The cluster manager schedules tasks on the cluster in local mode.

C. The cluster manager allocates resources to Spark applications and maintains the executor processes in client mode.

D. The cluster manager allocates resources to Spark applications and maintains the executor processes in remote mode.

E. The cluster manager allocates resources to the DataFrame manager.

Correct Answer: C

The cluster manager allocates resources to Spark applications and maintains the executor processes in client mode. Correct. In cluster mode, the cluster manager is located on a node other than the client machine. From there it starts and ends executor processes on the cluster nodes as required by the Spark application running on the Spark driver. The cluster manager allocates resources to Spark applications and maintains the executor processes in remote mode. Wrong, there is no "remote" execution mode in Spark. Available execution modes are local, client, and cluster. The cluster manager allocates resources to the DataFrame manager Wrong, there is no "DataFrame manager" in Spark. The cluster manager schedules tasks on the cluster in client mode. No, in client mode, the Spark driver schedules tasks on the cluster ?not the cluster manager. The cluster manager schedules tasks on the cluster in local mode. Wrong: In local mode, there is no "cluster". The Spark application is running on a single machine, not on a cluster of machines.

QUESTION 4

Which of the following statements about executors is correct?

A. Executors are launched by the driver.

B. Executors stop upon application completion by default.

C. Each node hosts a single executor.

D. Executors store data in memory only.

E. An executor can serve multiple applications.

Correct Answer: B

QUESTION 5

Which of the following code blocks returns a 2-column DataFrame that shows the distinct values in column productId and the number of rows with that productId in DataFrame transactionsDf?

A. transactionsDf.count("productId").distinct()

B. transactionsDf.groupBy("productId").agg(col("value").count())

C. transactionsDf.count("productId")

D. transactionsDf.groupBy("productId").count()

E. transactionsDf.groupBy("productId").select(count("value"))

Correct Answer: D

transactionsDf.groupBy("productId").count()

Correct. This code block first groups DataFrame transactionsDf by column productId and then counts the rows in each group.

transactionsDf.groupBy("productId").select(count("value")) Incorrect. You cannot call select on a GroupedData object (the output of a groupBy) statement.

transactionsDf.count("productId")

No. DataFrame.count() does not take any arguments.

transactionsDf.count("productId").distinct()

Wrong. Since DataFrame.count() does not take any arguments, this option cannot be right.

transactionsDf.groupBy("productId").agg(col("value").count()) False. A Column object, as returned by col ("value"), does not have a count() method. You can see all available methods for Column object linked in the Spark documentation below. More info: pyspark.sql.DataFrame.count -- PySpark 3.1.2 documentation, pyspark.sql.Column -- PySpark 3.1.2 documentation

Static notebook | Dynamic notebook: See test 3, 41 (Databricks import instructions)