# 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**

The code block shown below should return a DataFrame with columns transactionsId, predError, value, and f from DataFrame transactionsDf. Choose the answer that correctly fills the blanks in the code block to accomplish this.

transactionsDf.__1__(__2__)

A. 1. filter

2. "transactionId", "predError", "value", "f"

B. 1. select

2. "transactionId, predError, value, f"

C. 1. select

2. ["transactionId", "predError", "value", "f"]

D. 1. where

2. col("transactionId"), col("predError"), col("value"), col("f") E. 1. select

2. col(["transactionId", "predError", "value", "f"])

Correct Answer: C

**QUESTION 2**

Which of the following code blocks silently writes DataFrame itemsDf in avro format to location fileLocation if a file does not yet exist at that location?

A. itemsDf.write.avro(fileLocation)

B. itemsDf.write.format("avro").mode("ignore").save(fileLocation)

C. itemsDf.write.format("avro").mode("errorifexists").save(fileLocation)

D. itemsDf.save.format("avro").mode("ignore").write(fileLocation)

E. spark.DataFrameWriter(itemsDf).format("avro").write(fileLocation)

Correct Answer: A

**QUESTION 3**

The code block shown below should add column transactionDateForm to DataFrame transactionsDf. The column

should express the unix-format timestamps in column transactionDate as string type like Apr 26 (Sunday). Choose the answer that correctly fills the blanks in the code block to accomplish this.

transactionsDf.__1__(__2__, from_unixtime(__3__, __4__))

A. 1. withColumn

2.

"transactionDateForm"

3.

"MMM d (EEEE)"

4.

"transactionDate"

B. 1. select

2.

"transactionDate"

3.

"transactionDateForm"

4.

"MMM d (EEEE)"

C. 1. withColumn

2.

"transactionDateForm"

3.

"transactionDate"

4.

"MMM d (EEEE)"

D. 1. withColumn

2.

"transactionDateForm"

3.

"transactionDate"

4.

"MM d (EEE)"

E. 1. withColumnRenamed

2.

"transactionDate"

3.

"transactionDateForm"

4.

"MM d (EEE)"

Correct Answer: C

**QUESTION 4**

Which of the following describes properties of a shuffle?

A. Operations involving shuffles are never evaluated lazily.

B. Shuffles involve only single partitions.

C. Shuffles belong to a class known as "full transformations".

D. A shuffle is one of many actions in Spark.

E. In a shuffle, Spark writes data to disk.

Correct Answer: E

In a shuffle, Spark writes data to disk.

Correct! Spark\\'s architecture dictates that intermediate results during a shuffle are written to disk.

A shuffle is one of many actions in Spark.

Incorrect. A shuffle is a transformation, but not an action.

Shuffles involve only single partitions.

No, shuffles involve multiple partitions. During a shuffle, Spark generates output partitions from multiple

input partitions.

Operations involving shuffles are never evaluated lazily. Wrong. A shuffle is a costly operation and Spark

will evaluate it as lazily as other transformations. This is, until a subsequent action triggers its evaluation.

Shuffles belong to a class known as "full transformations". Not quite. Shuffles belong to a class known as

"wide transformations". "Full transformation" is not a relevant term in Spark.

More info: Spark ?The Definitive Guide, Chapter 2 and Spark: disk I/O on stage boundaries explanation Stack Overflow

---

**QUESTION 5**

The code block displayed below contains an error. The code block should read the csv file located at path data/transactions.csv into DataFrame transactionsDf, using the first row as column header and casting the columns in the most appropriate type. Find the error. First 3 rows of transactions.csv: 1.transactionId;storeId;productId;name 2.1;23;12;green grass 3.2;35;31;yellow sun 4.3;23;12;green grass Code block: transactionsDf = spark.read.load("data/transactions.csv", sep=";", format="csv", header=True)

A. The DataFrameReader is not accessed correctly.

B. The transaction is evaluated lazily, so no file will be read.

C. Spark is unable to understand the file type.

D. The code block is unable to capture all columns.

E. The resulting DataFrame will not have the appropriate schema.

Correct Answer: E

Correct code block:

transactionsDf = spark.read.load("data/transactions.csv", sep=";", format="csv", header=True,

inferSchema=True)

By default, Spark does not infer the schema of the CSV (since this usually takes some time). So, you need

to add the inferSchema=True option to the code block.

More info: pyspark.sql.DataFrameReader.csv -- PySpark 3.1.2 documentation