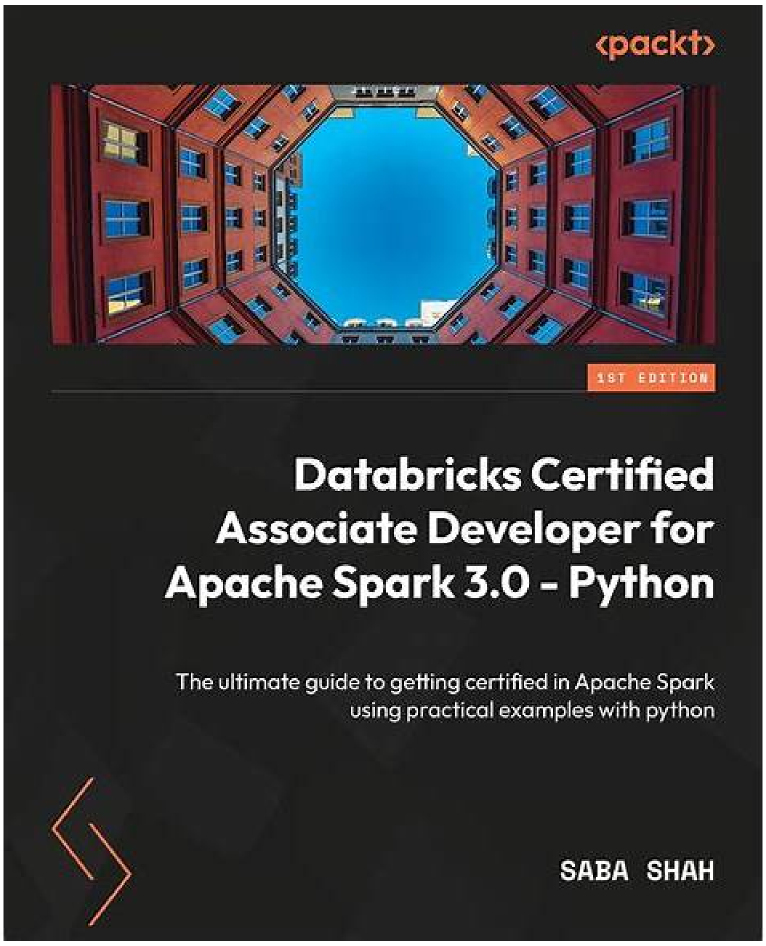
Test Associate-Developer-Apache-Spark Cram Pdf & New Associate-Developer-Apache-Spark Test Discount


‹packt›
1ST EDITION
Databricks Certified Associate Developer for Apache Spark 3.0 - Python
The ultimate guide to getting certified in Apache Spark using practical examples with python
SABA SHAH

BONUS!!! Download part of Exam4Free Associate-Developer-Apache-Spark dumps for free: https://drive.google.com/open?id=1y82JVK0ll4LqaKAbodwBHQCKdUBudM9l

At present, our Associate-Developer-Apache-Spark exam guide gains popularity in the market. The quality of our Associate-Developer-Apache-Spark training material is excellent. After all, we have undergone about ten years' development. Never has our practice test let customers down. Although we also face many challenges and troubles, our company get over them successfully. If you are determined to learn some useful skills, our Associate-Developer-Apache-Spark Real Dumps will be your good assistant. Then you will seize the good chance rather than others.

Importance of Databricks Associate Developer Apache Spark Exam to Secure your future

Data science is a rapidly growing field that's being used in a wide range of industries today. It's an essential skill for any developer or business person looking to make the most out of their data. Whether you want to be a part of the rapidly expanding world of data science or are just starting out with your career, the Data Science Associate Developer exam is the best way to test your knowledge and skills in this field. **Databricks Associate Developer Apache Spark exam dumps** are the best way to prepare for this exam.

In the current market scenario, the demand for Data Scientist is increasing day by day. As a Data Scientist, you are required to analyze the data and predict the future trends. The only thing that you need to do is to find out the right data and use the right tools. There are various tools that are available for you to perform data analysis and you can choose the one that suits your needs. Some of the tools that are used in data analysis are R, Python, Tableau, SAS, etc. Data Scientists are required to work on many different platforms and tools and they should be able to switch between them easily.

>> Test Associate-Developer-Apache-Spark Cram Pdf <<

Databricks Associate-Developer-Apache-Spark Exam Questions Are Out - Download And Prepare [2025]

The Associate-Developer-Apache-Spark examination time is approaching. Faced with a lot of learning content, you may be confused and do not know where to start. Associate-Developer-Apache-Spark study materials simplify the complex concepts and add examples, simulations, and diagrams to explain anything that may be difficult to understand. You can more easily master and simplify important test sites with Associate-Developer-Apache-Spark study materials. In addition, are you still feeling uncomfortable about giving up a lot of time to entertain, work or accompany your family and friends in preparation for the exam? Using Associate-Developer-Apache-Spark Learning Materials, you can spend less time and effort reviewing and preparing, which will help you save a lot of time and energy. Then you can do whatever you want. Actually, if you can guarantee that your effective learning time with Associate-Developer-Apache-Spark study materials is up to 20-30 hours, you can pass the exam.

Databricks Certified Associate Developer for Apache Spark 3.0 is a certification exam that validates the skills and knowledge of individuals in developing Apache Spark applications using Databricks. Associate-Developer-Apache-Spark Exam is designed to test the understanding of fundamental Spark concepts, programming in Spark using Python or Scala, manipulating data using Spark, and testing and debugging Spark applications.

Databricks Associate-Developer-Apache-Spark certification exam is a valuable credential for developers and data engineers looking to demonstrate their expertise in Apache Spark development using Databricks. Databricks Certified Associate Developer for Apache Spark 3.0 Exam certification exam covers a wide range of topics related to Apache Spark and is designed to test the candidate's knowledge and ability to develop and deploy Apache Spark applications using Databricks. Passing the exam provides candidates with a digital badge and certificate that can be displayed on their professional profiles and helps them stand out in the job market.

Databricks Certified Associate Developer for Apache Spark 3.0 Exam Sample Questions (Q30-Q35):

NEW QUESTION # 30
Which of the following code blocks returns only rows from DataFrame transactionsDf in which values in column productId are unique?

- A. transactionsDf.distinct("productId")
- B. transactionsDf.drop_duplicates(subset="productId")
- C. transactionsDf.unique("productId")
- D. transactionsDf.dropDuplicates(subset="productId")
- E. transactionsDf.dropDuplicates(subset=["productId"])

Answer: E

Explanation:
Explanation
Although the question suggests using a method called unique() here, that method does not actually exist in PySpark. In PySpark, it is called distinct(). But then, this method is not the right one to use here, since with distinct() we could filter out unique values in a specific column.
However, we want to return the entire rows here. So the trick is to use dropDuplicates with the subset keyword parameter. In the documentation for dropDuplicates, the examples show that subset should be used with a list. And this is exactly the key to solving this question: The productId column needs to be fed into the subset argument in a list, even though it is just a single column.
More info: pyspark.sql.DataFrame.dropDuplicates - PySpark 3.1.1 documentation Static notebook | Dynamic notebook: See test 1

NEW QUESTION # 31
Which of the following code blocks writes DataFrame itemsDf to disk at storage location filePath, making sure to substitute any existing data at that location?

- A. itemsDf.write.mode("overwrite").parquet(filePath)
- B. itemsDf.write(filePath, mode="overwrite")
- C. itemsDf.write.option("parquet").mode("overwrite").path(filePath)

- D. itemsDf.write.mode("overwrite").path(filePath)
- E. itemsDf.write().parquet(filePath, mode="overwrite")

Answer: A

Explanation:
Explanation
itemsDf.write.mode("overwrite").parquet(filePath)
Correct! itemsDf.write returns a pyspark.sql.DataFrameWriter instance whose overwriting behavior can be modified via the mode setting or by passing mode="overwrite" to the parquet() command.
Although the parquet format is not prescribed for solving this question, parquet() is a valid operator to initiate Spark to write the data to disk.
itemsDf.write.mode("overwrite").path(filePath)
No. A pyspark.sql.DataFrameWriter instance does not have a path() method.
itemsDf.write.option("parquet").mode("overwrite").path(filePath)
Incorrect, see above. In addition, a file format cannot be passed via the option() method.
itemsDf.write(filePath, mode="overwrite")
Wrong. Unfortunately, this is too simple. You need to obtain access to a DataFrameWriter for the DataFrame through calling itemsDf.write upon which you can apply further methods to control how Spark data should be written to disk. You cannot, however, pass arguments to itemsDf.write directly.
itemsDf.write().parquet(filePath, mode="overwrite")
False. See above.
More info: pyspark.sql.DataFrameWriter.parquet - PySpark 3.1.2 documentation Static notebook | Dynamic notebook: See test 3

NEW QUESTION # 32
The code block displayed below contains an error. The code block should display the schema of DataFrame transactionsDf. Find the error.
Code block:
transactionsDf.rdd.printSchema

- A. printSchema is a method and should be written as printSchema(). It is also not callable through transactionsDf.rdd, but should be called directly from transactionsDf.
(Correct)
- B. There is no way to print a schema directly in Spark, since the schema can be printed easily through using print(transactionsDf.columns), so that should be used instead.
- C. printSchema is only accessible through the spark session, so the code block should be rewritten as spark.printSchema(transactionsDf).
- D. printSchema is a not a method of transactionsDf.rdd. Instead, the schema should be printed via transactionsDf.print_schema().
- E. The code block should be wrapped into a print() operation.

Answer: A

Explanation:
Explanation
Correct code block:
transactionsDf.printSchema()
This is more of a knowledge question that you should just memorize or look up in the provided documentation during the exam. You can get more info about DataFrame.printSchema() in the documentation (link below). However - it is a plain simple method without any arguments.
One answer points to an alternative of printing the schema: You could also use print(transactionsDf.schema).
This will give you readable, but not nicely formatted, description of the schema.
More info: pyspark.sql.DataFrame.printSchema - PySpark 3.1.1 documentation Static notebook | Dynamic notebook: See test 1

NEW QUESTION # 33
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"))
itemsDf.withColumnRenamed("itemNameElements", split("itemName"))

- A. Operator withColumnRenamed needs to be replaced with operator withColumn and the split method needs to be replaced by the splitString method.
- B. The expressions "itemNameElements" and split("itemName") need to be swapped.
- C. Operator withColumnRenamed needs to be replaced with operator withColumn and a second argument " " needs to be passed to the split method.
- D. All column names need to be wrapped in the col() operator.
- E. Operator withColumnRenamed needs to be replaced with operator withColumn and a second argument "," needs to be passed to the split method.

Answer: C

Explanation:
Explanation
Correct code block:
itemsDf.withColumn("itemNameElements", split("itemName"," "))
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]
3	Outdoors Backpack	Sports Company Inc.	[Outdoors, Backpack]
+------+---------------------------------+------------------+---+ The key to solving this question is that the split method definitely needs a second argument here (also look at the link to the documentation below). Given the values in column itemName in DataFrame itemsDf, this should be a space character " ". This is the character we need to split the words in the column.
More info: pyspark.sql.functions.split - PySpark 3.1.1 documentation
Static notebook | Dynamic notebook: See test 1

NEW QUESTION # 34
In which order should the code blocks shown below be run in order to read a JSON file from location jsonPath into a DataFrame and return only the rows that do not have value 3 in column productId?
1. importedDf.createOrReplaceTempView("importedDf")
2. spark.sql("SELECT * FROM importedDf WHERE productId != 3")
3. spark.sql("FILTER * FROM importedDf WHERE productId != 3")
4. importedDf = spark.read.option("format", "json").path(jsonPath)
5. importedDf = spark.read.json(jsonPath)

- A. 4, 1, 2
- B. 5, 2
- C. 5, 1, 3
- D. 4, 1, 3
- E. 5, 1, 2

Answer: E

Explanation:
Explanation
Correct code block:
importedDf = spark.read.json(jsonPath)

```
importedDf.createOrReplaceTempView("importedDf")
spark.sql("SELECT * FROM importedDf WHERE productId != 3")
```

Option 5 is the only correct way listed of reading in a JSON in PySpark. The option("format", "json") is not the correct way to tell Spark's DataFrameReader that you want to read a JSON file. You would do this through format("json") instead. Also, you can communicate the specific path of the JSON file to the DataFramReader using the load() method, not the path() method.
In order to use a SQL command through the SparkSession spark, you first need to create a temporary view through DataFrame.createOrReplaceTempView().
The SQL statement should start with the SELECT operator. The FILTER operator SQL provides is not the correct one to use here.
Static notebook | Dynamic notebook: See test 2

NEW QUESTION # 35
......