PassTest

Bessere Qualität , bessere Dienstleistungen!

Q&A

http://www.passtest.de

Einjährige kostenlose Aktualisierung

Exam : **Databricks Certified Professional Data Engineer**

Title : Databricks Certified Data Engineer Professional Exam

Version : DEMO

1.An upstream system has been configured to pass the date for a given batch of data to the Databricks Jobs API as a parameter.
The notebook to be scheduled will use this parameter to load data with the following code:
df = spark.read.format("parquet").load(f"/mnt/source/(date)")
Which code block should be used to create the date Python variable used in the above code block?
A. date = spark.conf.get("date")
B. input_dict = input()
date= input_dict["date"]
C. import sys
date = sys.argv[1]
D. date = dbutils.notebooks.getParam("date")
E. dbutils.widgets.text("date", "null")
date = dbutils.widgets.get("date")
Answer: E
Explanation:
The code block that should be used to create the date Python variable used in the above code block is: dbutils.widgets.text(“date”, “null”) date = dbutils.widgets.get(“date”)
This code block uses the dbutils.widgets API to create and get a text widget named “date” that can accept a string value as a parameter1. The default value of the widget is “null”, which means that if no parameter is passed, the date variable will be “null”. However, if a parameter is passed through the Databricks Jobs API, the date variable will be assigned the value of the parameter. For example, if the parameter is “2021-11-01”, the date variable will be “2021-11-01”. This way, the notebook can use the date variable to load data from the specified path. The other options are not correct, because:
Option A is incorrect because spark.conf.get(“date”) is not a valid way to get a parameter passed through the Databricks Jobs API. The spark.conf API is used to get or set Spark configuration properties, not notebook parameters2.
Option B is incorrect because input() is not a valid way to get a parameter passed through the Databricks Jobs API. The input() function is used to get user input from the standard input stream, not from the API request3.
Option C is incorrect because sys.argv1 is not a valid way to get a parameter passed through the Databricks Jobs API. The sys.argv list is used to get the command-line arguments passed to a Python script, not to a notebook4.
Option D is incorrect because dbutils.notebooks.getParam(“date”) is not a valid way to get a parameter passed through the Databricks Jobs API. The dbutils.notebooks API is used to get or set notebook parameters when running a notebook as a job or as a subnotebook, not when passing parameters through the API5.
Reference: Widgets, Spark Configuration, input(), sys.argv, Notebooks

2.The Databricks workspace administrator has configured interactive clusters for each of the data engineering groups. To control costs, clusters are set to terminate after 30 minutes of inactivity. Each user should be able to execute workloads against their assigned clusters at any time of the day.
Assuming users have been added to a workspace but not granted any permissions, which of the following describes the minimal permissions a user would need to start and attach to an already configured cluster.

A. "Can Manage" privileges on the required cluster
B. Workspace Admin privileges, cluster creation allowed. "Can Attach To" privileges on the required cluster
C. Cluster creation allowed. "Can Attach To" privileges on the required cluster
D. "Can Restart" privileges on the required cluster
E. Cluster creation allowed. "Can Restart" privileges on the required cluster
Answer: D
Explanation:
https://learn.microsoft.com/en-us/azure/databricks/security/auth-authz/access-control/cluster-acl
https://docs.databricks.com/en/security/auth-authz/access-control/cluster-acl.html

3.When scheduling Structured Streaming jobs for production, which configuration automatically recovers from query failures and keeps costs low?
A. Cluster: New Job Cluster;
Retries: Unlimited;
Maximum Concurrent Runs: Unlimited
B. Cluster: New Job Cluster;
Retries: None;
Maximum Concurrent Runs: 1
C. Cluster: Existing All-Purpose Cluster; Retries: Unlimited;
Maximum Concurrent Runs: 1
D. Cluster: Existing All-Purpose Cluster; Retries: Unlimited;
Maximum Concurrent Runs: 1
E. Cluster: Existing All-Purpose Cluster; Retries: None;
Maximum Concurrent Runs: 1
Answer: D
Explanation:
The configuration that automatically recovers from query failures and keeps costs low is to use a new job cluster, set retries to unlimited, and set maximum concurrent runs to 1.
This configuration has the following advantages:
A new job cluster is a cluster that is created and terminated for each job run. This means that the cluster resources are only used when the job is running, and no idle costs are incurred. This also ensures that the cluster is always in a clean state and has the latest configuration and libraries for the job1.
Setting retries to unlimited means that the job will automatically restart the query in case of any failure, such as network issues, node failures, or transient errors. This improves the reliability and availability of the streaming job, and avoids data loss or inconsistency2.
Setting maximum concurrent runs to 1 means that only one instance of the job can run at a time. This prevents multiple queries from competing for the same resources or writing to the same output location, which can cause performance degradation or data corruption3.
Therefore, this configuration is the best practice for scheduling Structured Streaming jobs for production, as it ensures that the job is resilient, efficient, and consistent.
Reference: Job clusters, Job retries, Maximum concurrent runs

4.The data engineering team has configured a Databricks SQL query and alert to monitor the values in

a Delta Lake table. The recent_sensor_recordings table contains an identifying sensor_id alongside the timestamp and temperature for the most recent 5 minutes of recordings.
The below query is used to create the alert:

```
SELECT MEAN(temperature), MAX(temperature), MIN(temperature)
FROM recent_sensor_recordings
GROUP BY sensor_id
```

The query is set to refresh each minute and always completes in less than 10 seconds. The alert is set to trigger when mean (temperature) > 120. Notifications are triggered to be sent at most every 1 minute.
If this alert raises notifications for 3 consecutive minutes and then stops, which statement must be true?
A. The total average temperature across all sensors exceeded 120 on three consecutive executions of the query
B. The recent_sensor_recordingstable was unresponsive for three consecutive runs of the query
C. The source query failed to update properly for three consecutive minutes and then restarted
D. The maximum temperature recording for at least one sensor exceeded 120 on three consecutive executions of the query
E. The average temperature recordings for at least one sensor exceeded 120 on three consecutive executions of the query
Answer: E
Explanation:
This is the correct answer because the query is using a GROUP BY clause on the sensor_id column, which means it will calculate the mean temperature for each sensor separately. The alert will trigger when the mean temperature for any sensor is greater than 120, which means at least one sensor had an average temperature above 120 for three consecutive minutes. The alert will stop when the mean temperature for all sensors drops below 120.
Verified Reference: [Databricks Certified Data Engineer Professional], under “SQL Analytics” section; Databricks Documentation, under “Alerts” section.

5.A junior developer complains that the code in their notebook isn't producing the correct results in the development environment. A shared screenshot reveals that while they're using a notebook versioned with Databricks Repos, they're using a personal branch that contains old logic. The desired branch named dev-2.3.9 is not available from the branch selection dropdown.
Which approach will allow this developer to review the current logic for this notebook?
A. Use Repos to make a pull request use the Databricks REST API to update the current branch to dev-2.3.9
B. Use Repos to pull changes from the remote Git repository and select the dev-2.3.9 branch.
C. Use Repos to checkout the dev-2.3.9 branch and auto-resolve conflicts with the current branch
D. Merge all changes back to the main branch in the remote Git repository and clone the repo again
E. Use Repos to merge the current branch and the dev-2.3.9 branch, then make a pull request to sync with the remote repository
Answer: B
Explanation:
This is the correct answer because it will allow the developer to update their local repository with the latest changes from the remote repository and switch to the desired branch. Pulling changes will not affect the current branch or create any conflicts, as it will only fetch the changes and not merge them.

Selecting the dev-2.3.9 branch from the dropdown will checkout that branch and display its contents in the notebook.
Verified Reference: [Databricks Certified Data Engineer Professional], under “Databricks Tooling” section; Databricks Documentation, under “Pull changes from a remote repository” section.