

100%합격보장가능한CDP-3002최신덤프샘플문제최신 버전덤프



Fast2test는 응시자에게 있어서 시간이 정말 소중한다는 것을 잘 알고 있으므로 Cloudera CDP-3002덤프를 자주 업데이트 하고, 오래 되고 더 이상 사용 하지 않는 문제들은 바로 삭제해버리며 새로운 최신 문제들을 추가 합니다. 이는 응시자가 확실하고도 빠르게Cloudera CDP-3002덤프를 마스터하고Cloudera CDP-3002시험을 패스할수 있도록 하는 또 하나의 보장입니다.

만약 아직도 우리를 선택할지에 대하여 망설이고 있다면. 우선은 우리 사이트에서 Fast2test가 제공하는 무료인 일부 문제와 답을 다운하여 체험해보시고 결정을 내리시길 바랍니다.그러면 우리의 덤프에 믿음;갈 것이고,우리 또한 우리의 문제와 답들은 무조건 100%통과 율로 아주 고득점으로Cloudera인증CDP-3002시험을 패스하실 수 있습니다,

>> CDP-3002최신 덤프샘플문제 <<

최신버전 CDP-3002최신 덤프샘플문제 완벽한 시험덤프

Fast2test의 제품들은 모두 우리만의 거대한IT업계엘리트들로 이루어진 그룹 즉 관련업계에서 권위가 있는 전문가들이 자기만의 지식과 지금까지의 경험으로 최고의 IT인증관련자료를 만들어냅니다. Fast2test의 문제와 답은 정확도 적용률이 아주 높습니다. 우리의 덤프로 완벽한Cloudera인증CDP-3002시험대비를 하시면 되겠습니다. 이렇게 어려운 시험은 우리Cloudera인증CDP-3002덤프로 여러분의 고민과 꿈을 한방에 해결해드립니다.

최신 Cloudera Certification CDP-3002 무료샘플문제 (Q316-Q321):

질문 # 316

You're working with a large DAG that contains numerous tasks and complex dependencies. How can you improve the DAG's readability and maintainability?

- A. Utilize comments sparingly within the DAG code, as the logic should be self-explanatory.
- **B. Break down the DAG into smaller sub-DAGs with well-defined functionalities and clear naming conventions.**
- C. Use cryptic and abbreviated names for tasks and variables, assuming everyone understands the context.
- D. Implement extensive logging within each task, regardless of its purpose, to capture detailed execution information.

정답: B

질문 # 317

You're working with a large dataset stored in multiple Parquet files across different HDFS directories. How can you efficiently load and process this data using Spark, ensuring data locality and minimizing shuffle operations?

- A. Directly load all files using `spark.read.parquet("/path/to/data/")`
- **B. Leverage Spark SQL catalogs and partition discovery**
- C. Use `spark.read.parquet("/path/to/data/")` with recursive directory listing
- D. Implement a custom function to read each Parquet file individually

정답: B

설명:

While options A and B might work, they don't optimize locality or shuffle. Option C is inefficient. By defining the data location and schema in a Spark SQL catalog, Spark can automatically discover partitions and efficiently read data in parallel, minimizing shuffle across the network.

질문 # 318

You're provisioning a new Cloudera Data Engineering (CDE) virtual cluster. Which of the following factors should you consider when choosing an appropriate instance type for Iceberg workloads? (Choose two)

- A. The complexity of your SQL queries and transformations
- B. Whether you require GPU acceleration for machine learning tasks integrated with Iceberg
- C. The type of storage (SSD vs. HDD) needed for optimal performance
- D. The expected size of your Iceberg tables
- E. The network bandwidth required by your Iceberg jobs

정답: C,D

설명:

The size of your tables directly influences the needed RAM and storage of the cluster nodes. Iceberg is optimized for fast reads and writes, especially with columnar file formats like Parquet. SSDs provide far superior random read/write performance compared to HDDs for Iceberg operations. While instance types affect query complexity capability, the key factors are table size and storage speed. Network bandwidth is less critical unless dealing with extremely distributed systems.

질문 # 319

What is the recommended approach in Apache Airflow for ensuring data quality checks are performed after data is loaded into multiple target systems, which might complete their loading processes at different times?

- A. Schedule the data quality checks at a fixed delay after the longest expected load time.
- B. Utilize a BranchPythonOperator to dynamically route the workflow based on system availability.
- C. Implement multiple ExternalTaskSensor instances, each waiting for a specific loading task to complete.
- D. Use a CrossDagDep operator to manage dependencies across multiple DAGs.

정답: C

설명:

The ExternalTaskSensor is designed to wait for a task in another DAG to complete before proceeding. By using multiple instances of this sensor, each configured to wait for data loading to complete in a different target system, you can ensure that data quality checks are only initiated once all target systems have successfully completed their loading processes. This approach allows for precise control and synchronization across disparate systems and processes.

질문 # 320

A PySpark application is facing performance issues due to uneven distribution of data across the nodes. Which approach would best help in resolving this issue?

- A. Use 'df.collect()'.
• B. Employ to redistribute data based on a key.
- C. Convert the DataFrame to an RDD and then back to a DataFrame.
- D. Apply 'df.coalesce(Y to reduce the number of partitions.

정답: B

설명:

Using repartition in PySpark helps to redistribute the data more evenly based on the specified key column. This can alleviate issues caused by data skewness and improve the performance of distributed operations.

