

# Amazon

## Exam Questions AWS-Certified-Data-Engineer-Associate

AWS Certified Data Engineer - Associate (DEA-C01)



**NEW QUESTION 1**

A data engineer maintains custom Python scripts that perform a data formatting process that many AWS Lambda functions use. When the data engineer needs to modify the Python scripts, the data engineer must manually update all the Lambda functions.

The data engineer requires a less manual way to update the Lambda functions. Which solution will meet this requirement?

- A. Store a pointer to the custom Python scripts in the execution context object in a shared Amazon S3 bucket.
- B. Package the custom Python scripts into Lambda layer
- C. Apply the Lambda layers to the Lambda functions.
- D. Store a pointer to the custom Python scripts in environment variables in a shared Amazon S3 bucket.
- E. Assign the same alias to each Lambda function
- F. Call each Lambda function by specifying the function's alias.

**Answer: B**

**Explanation:**

Lambda layers are a way to share code and dependencies across multiple Lambda functions. By packaging the custom Python scripts into Lambda layers, the data engineer can update the scripts in one place and have them automatically applied to all the Lambda functions that use the layer. This reduces the manual effort and ensures consistency across the Lambda functions. The other options are either not feasible or not efficient. Storing a pointer to the custom Python scripts in the execution context object or in environment variables would require the Lambda functions to download the scripts from Amazon S3 every time they are invoked, which would increase latency and cost. Assigning the same alias to each Lambda function would not help with updating the Python scripts, as the alias only points to a specific version of the Lambda function code. References:

? AWS Lambda layers

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 3: Data Ingestion and Transformation, Section 3.4: AWS Lambda

**NEW QUESTION 2**

A company maintains an Amazon Redshift provisioned cluster that the company uses for extract, transform, and load (ETL) operations to support critical analysis tasks. A sales team within the company maintains a Redshift cluster that the sales team uses for business intelligence (BI) tasks.

The sales team recently requested access to the data that is in the ETL Redshift cluster so the team can perform weekly summary analysis tasks. The sales team needs to join data from the ETL cluster with data that is in the sales team's BI cluster.

The company needs a solution that will share the ETL cluster data with the sales team without interrupting the critical analysis tasks. The solution must minimize usage of the computing resources of the ETL cluster.

Which solution will meet these requirements?

- A. Set up the sales team BI cluster as a consumer of the ETL cluster by using Redshift data sharing.
- B. Create materialized views based on the sales team's requirement
- C. Grant the sales team direct access to the ETL cluster.
- D. Create database views based on the sales team's requirement
- E. Grant the sales team direct access to the ETL cluster.
- F. Unload a copy of the data from the ETL cluster to an Amazon S3 bucket every week
- G. Create an Amazon Redshift Spectrum table based on the content of the ETL cluster.

**Answer: A**

**Explanation:**

Redshift data sharing is a feature that enables you to share live data across different Redshift clusters without the need to copy or move data. Data sharing provides secure and governed access to data, while preserving the performance and concurrency benefits of Redshift. By setting up the sales team BI cluster as a consumer of the ETL cluster, the company can share the ETL cluster data with the sales team without interrupting the critical analysis tasks. The solution also minimizes the usage of the computing resources of the ETL cluster, as the data sharing does not consume any storage space or compute resources from the producer cluster. The other options are either not feasible or not efficient. Creating materialized views or database views would require the sales team to have direct access to the ETL cluster, which could interfere with the critical analysis tasks. Unloading a copy of the data from the ETL cluster to an Amazon S3 bucket every week would introduce additional latency and cost, as well as create data inconsistency issues. References:

? Sharing data across Amazon Redshift clusters

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 2: Data Store Management, Section 2.2: Amazon Redshift

**NEW QUESTION 3**

A company stores petabytes of data in thousands of Amazon S3 buckets in the S3 Standard storage class. The data supports analytics workloads that have unpredictable and variable data access patterns.

The company does not access some data for months. However, the company must be able to retrieve all data within milliseconds. The company needs to optimize S3 storage costs.

Which solution will meet these requirements with the LEAST operational overhead?

- A. Use S3 Storage Lens standard metrics to determine when to move objects to more cost-optimized storage classes
- B. Create S3 Lifecycle policies for the S3 buckets to move objects to cost-optimized storage classes
- C. Continue to refine the S3 Lifecycle policies in the future to optimize storage costs.
- D. Use S3 Storage Lens activity metrics to identify S3 buckets that the company accesses infrequently
- E. Configure S3 Lifecycle rules to move objects from S3 Standard to the S3 Standard-Infrequent Access (S3 Standard-IA) and S3 Glacier storage classes based on the age of the data.
- F. Use S3 Intelligent-Tiering
- G. Activate the Deep Archive Access tier.
- H. Use S3 Intelligent-Tiering
- I. Use the default access tier.

**Answer: D**

**Explanation:**

S3 Intelligent-Tiering is a storage class that automatically moves objects between four access tiers based on the changing access patterns. The default access tier consists of two tiers: Frequent Access and Infrequent Access. Objects in the Frequent Access tier have the same performance and availability as S3 Standard, while objects in the Infrequent Access tier have the same performance and availability as S3 Standard-IA. S3 Intelligent-Tiering monitors the access patterns of each object and moves them between the tiers accordingly, without any operational overhead or retrieval fees. This solution can optimize S3 storage costs for data

with unpredictable and variable access patterns, while ensuring millisecond latency for data retrieval. The other solutions are not optimal or relevant for this requirement. Using S3 Storage Lens standard metrics and activity metrics can provide insights into the storage usage and access patterns, but they do not automate the data movement between storage classes. Creating S3 Lifecycle policies for the S3 buckets can move objects to more cost-optimized storage classes, but they require manual configuration and maintenance, and they may incur retrieval fees for data that is accessed unexpectedly. Activating the Deep Archive Access tier for S3 Intelligent-Tiering can further reduce the storage costs for data that is rarely accessed, but it also increases the retrieval time to 12 hours, which does not meet the requirement of millisecond latency. References:

- ? S3 Intelligent-Tiering
- ? S3 Storage Lens
- ? S3 Lifecycle policies
- ? [AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide]

#### NEW QUESTION 4

A data engineer must manage the ingestion of real-time streaming data into AWS. The data engineer wants to perform real-time analytics on the incoming streaming data by using time-based aggregations over a window of up to 30 minutes. The data engineer needs a solution that is highly fault tolerant. Which solution will meet these requirements with the LEAST operational overhead?

- A. Use an AWS Lambda function that includes both the business and the analytics logic to perform time-based aggregations over a window of up to 30 minutes for the data in Amazon Kinesis Data Streams.
- B. Use Amazon Managed Service for Apache Flink (previously known as Amazon Kinesis Data Analytics) to analyze the data that might occasionally contain duplicates by using multiple types of aggregations.
- C. Use an AWS Lambda function that includes both the business and the analytics logic to perform aggregations for a tumbling window of up to 30 minutes, based on the event timestamp.
- D. Use Amazon Managed Service for Apache Flink (previously known as Amazon Kinesis Data Analytics) to analyze the data by using multiple types of aggregations to perform time-based analytics over a window of up to 30 minutes.

**Answer:** A

#### Explanation:

This solution meets the requirements of managing the ingestion of real-time streaming data into AWS and performing real-time analytics on the incoming streaming data with the least operational overhead. Amazon Managed Service for Apache Flink is a fully managed service that allows you to run Apache Flink applications without having to manage any infrastructure or clusters. Apache Flink is a framework for stateful stream processing that supports various types of aggregations, such as tumbling, sliding, and session windows, over streaming data. By using Amazon Managed Service for Apache Flink, you can easily connect to Amazon Kinesis Data Streams as the source and sink of your streaming data, and perform time-based analytics over a window of up to 30 minutes. This solution is also highly fault tolerant, as Amazon Managed Service for Apache Flink automatically scales, monitors, and restarts your Flink applications in case of failures. References:

- ? Amazon Managed Service for Apache Flink
- ? Apache Flink
- ? Window Aggregations in Flink

#### NEW QUESTION 5

A data engineer needs to use an Amazon QuickSight dashboard that is based on Amazon Athena queries on data that is stored in an Amazon S3 bucket. When the data engineer connects to the QuickSight dashboard, the data engineer receives an error message that indicates insufficient permissions. Which factors could cause the permissions-related errors? (Choose two.)

- A. There is no connection between QuickSight and Athena.
- B. The Athena tables are not cataloged.
- C. QuickSight does not have access to the S3 bucket.
- D. QuickSight does not have access to decrypt S3 data.
- E. There is no IAM role assigned to QuickSight.

**Answer:** CD

#### Explanation:

QuickSight does not have access to the S3 bucket and QuickSight does not have access to decrypt S3 data are two possible factors that could cause the permissions-related errors. Amazon QuickSight is a business intelligence service that allows you to create and share interactive dashboards based on various data sources, including Amazon Athena. Amazon Athena is a serverless query service that allows you to analyze data stored in Amazon S3 using standard SQL. To use an Amazon QuickSight dashboard that is based on Amazon Athena queries on data that is stored in an Amazon S3 bucket, you need to grant QuickSight access to both Athena and S3, as well as any encryption keys that are used to encrypt the S3 data. If QuickSight does not have access to the S3 bucket or the encryption keys, it will not be able to read the data from Athena and display it on the dashboard, resulting in an error message that indicates insufficient permissions.

The other options are not factors that could cause the permissions-related errors. Option A, there is no connection between QuickSight and Athena, is not a factor, as QuickSight supports Athena as a native data source, and you can easily create a connection between them using the QuickSight console or the API. Option B, the Athena tables are not cataloged, is not a factor, as QuickSight can automatically discover the Athena tables that are cataloged in the AWS Glue Data Catalog, and you can also manually specify the Athena tables that are not cataloged. Option E, there is no IAM role assigned to QuickSight, is not a factor, as QuickSight requires an IAM role to access any AWS data sources, including Athena and S3, and you can create and assign an IAM role to QuickSight using the QuickSight console or the API. References:

- ? Using Amazon Athena as a Data Source
- ? Granting Amazon QuickSight Access to AWS Resources
- ? Encrypting Data at Rest in Amazon S3

#### NEW QUESTION 6

A company uses Amazon Athena to run SQL queries for extract, transform, and load (ETL) tasks by using Create Table As Select (CTAS). The company must use Apache Spark instead of SQL to generate analytics. Which solution will give the company the ability to use Spark to access Athena?

- A. Athena query settings
- B. Athena workgroup
- C. Athena data source
- D. Athena query editor

**Answer:** C

**Explanation:**

Athena data source is a solution that allows you to use Spark to access Athena by using the Athena JDBC driver and the Spark SQL interface. You can use the Athena data source to create Spark DataFrames from Athena tables, run SQL queries on the DataFrames, and write the results back to Athena. The Athena data source supports various data formats, such as CSV, JSON, ORC, and Parquet, and also supports partitioned and bucketed tables. The Athena data source is a cost-effective and scalable way to use Spark to access Athena, as it does not require any additional infrastructure or services, and you only pay for the data scanned by Athena.

The other options are not solutions that give the company the ability to use Spark to access Athena. Option A, Athena query settings, is a feature that allows you to configure various parameters for your Athena queries, such as the output location, the encryption settings, the query timeout, and the workgroup. Option B, Athena workgroup, is a feature that allows you to isolate and manage your Athena queries and resources, such as the query history, the query notifications, the query concurrency, and the query cost. Option D, Athena query editor, is a feature that allows you to write and run SQL queries on Athena using the web console or the API. None of these options enable you to use Spark instead of SQL to generate analytics on Athena. References:

? Using Apache Spark in Amazon Athena

? Athena JDBC Driver

? Spark SQL

? Athena query settings

? [Athena workgroups]

? [Athena query editor]

**NEW QUESTION 7**

A company is migrating its database servers from Amazon EC2 instances that run Microsoft SQL Server to Amazon RDS for Microsoft SQL Server DB instances. The company's analytics team must export large data elements every day until the migration is complete. The data elements are the result of SQL joins across multiple tables. The data must be in Apache Parquet format. The analytics team must store the data in Amazon S3.

Which solution will meet these requirements in the MOST operationally efficient way?

A. Create a view in the EC2 instance-based SQL Server databases that contains the required data element

B. Create an AWS Glue job that selects the data directly from the view and transfers the data in Parquet format to an S3 bucket

C. Schedule the AWS Glue job to run every day.

D. Schedule SQL Server Agent to run a daily SQL query that selects the desired data elements from the EC2 instance-based SQL Server database

E. Configure the query to direct the output .csv objects to an S3 bucket

F. Create an S3 event that invokes an AWS Lambda function to transform the output format from .csv to Parquet.

G. Use a SQL query to create a view in the EC2 instance-based SQL Server databases that contains the required data element

H. Create and run an AWS Glue crawler to read the view

I. Create an AWS Glue job that retrieves the data and transfers the data in Parquet format to an S3 bucket

J. Schedule the AWS Glue job to run every day.

K. Create an AWS Lambda function that queries the EC2 instance-based databases by using Java Database Connectivity (JDBC). Configure the Lambda function to retrieve the required data, transform the data into Parquet format, and transfer the data into an S3 bucket

L. Use Amazon EventBridge to schedule the Lambda function to run every day.

**Answer:** A

**Explanation:**

Option A is the most operationally efficient way to meet the requirements because it minimizes the number of steps and services involved in the data export process. AWS Glue is a fully managed service that can extract, transform, and load (ETL) data from various sources to various destinations, including Amazon S3. AWS Glue can also convert data to different formats, such as Parquet, which is a columnar storage format that is optimized for analytics. By creating a view in the SQL Server databases that contains the required data elements, the AWS Glue job can select the data directly from the view without having to perform any joins or transformations on the source data. The AWS Glue job can then transfer the data in Parquet format to an S3 bucket and run on a daily schedule.

Option B is not operationally efficient because it involves multiple steps and services to export the data. SQL Server Agent is a tool that can run scheduled tasks on SQL Server databases, such as executing SQL queries. However, SQL Server Agent cannot directly export data to S3, so the query output must be saved as .csv objects on the EC2 instance. Then, an S3 event must be configured to trigger an AWS Lambda function that can transform the .csv objects to Parquet format and upload them to S3. This option adds complexity and latency to the data export process and requires additional resources and configuration.

Option C is not operationally efficient because it introduces an unnecessary step of running an AWS Glue crawler to read the view. An AWS Glue crawler is a service that can scan data sources and create metadata tables in the AWS Glue Data Catalog. The Data Catalog is a central repository that stores information about the data sources, such as schema, format, and location. However, in this scenario, the schema and format of the data elements are already known and fixed, so there is no need to run a crawler to discover them. The AWS Glue job can directly select the data from the view without using the Data Catalog. Running a crawler adds extra time and cost to the data export process.

Option D is not operationally efficient because it requires custom code and configuration to query the databases and transform the data. An AWS Lambda function is a service that can run code in response to events or triggers, such as Amazon EventBridge. Amazon EventBridge is a service that can connect applications and services with event sources, such as schedules, and route them to targets, such as Lambda functions. However, in this scenario, using a Lambda function to query the databases and transform the data is not the best option because it requires writing and maintaining code that uses JDBC to connect to the SQL Server databases, retrieve the required data, convert the data to Parquet format, and transfer the data to S3. This option also has limitations on the execution time, memory, and concurrency of the Lambda function, which may affect the performance and reliability of the data export process.

References:

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

? AWS Glue Documentation

? Working with Views in AWS Glue

? Converting to Columnar Formats

**NEW QUESTION 8**

A company needs to partition the Amazon S3 storage that the company uses for a data lake. The partitioning will use a path of the S3 object keys in the following format: s3://bucket/prefix/year=2023/month=01/day=01.

A data engineer must ensure that the AWS Glue Data Catalog synchronizes with the S3 storage when the company adds new partitions to the bucket.

Which solution will meet these requirements with the LEAST latency?

A. Schedule an AWS Glue crawler to run every morning.

B. Manually run the AWS Glue CreatePartition API twice each day.

C. Use code that writes data to Amazon S3 to invoke the Boto3 AWS Glue create partition API call.

D. Run the MSCK REPAIR TABLE command from the AWS Glue console.

**Answer:** C



**Explanation:**

The best solution to ensure that the AWS Glue Data Catalog synchronizes with the S3 storage when the company adds new partitions to the bucket with the least latency is to use code that writes data to Amazon S3 to invoke the Boto3 AWS Glue create partition API call. This way, the Data Catalog is updated as soon as new data is written to S3, and the partition information is immediately available for querying by other services. The Boto3 AWS Glue create partition API call allows you to create a new partition in the Data Catalog by specifying the table name, the database name, and the partition values<sup>1</sup>. You can use this API call in your code that writes data to S3, such as a Python script or an AWS Glue ETL job, to create a partition for each new S3 object key that matches the partitioning scheme.

Option A is not the best solution, as scheduling an AWS Glue crawler to run every morning would introduce a significant latency between the time new data is written to S3 and the time the Data Catalog is updated. AWS Glue crawlers are processes that connect to a data store, progress through a prioritized list of classifiers to determine the schema for your data, and then create metadata tables in the Data Catalog<sup>2</sup>. Crawlers can be scheduled to run periodically, such as daily or hourly, but they cannot run continuously or in real-time. Therefore, using a crawler to synchronize the Data Catalog with the S3 storage would not meet the requirement of the least latency.

Option B is not the best solution, as manually running the AWS Glue CreatePartition API twice each day would also introduce a significant latency between the time new data is written to S3 and the time the Data Catalog is updated. Moreover, manually running the API would require more operational overhead and human intervention than using code that writes data to S3 to invoke the API automatically.

Option D is not the best solution, as running the MSCK REPAIR TABLE command from the AWS Glue console would also introduce a significant latency between the time new data is written to S3 and the time the Data Catalog is updated. The MSCK REPAIR TABLE command is a SQL command that you can run in the AWS Glue console to add partitions to the Data Catalog based on the S3 object keys that match the partitioning scheme<sup>3</sup>. However, this command is not meant to be run frequently or in real-time, as it can take a long time to scan the entire S3 bucket and add the partitions. Therefore, using this command to synchronize the Data Catalog with the S3 storage would not meet the requirement of the least latency. References:

? AWS Glue CreatePartition API

? Populating the AWS Glue Data Catalog

? MSCK REPAIR TABLE Command

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

**NEW QUESTION 9**

A data engineer must orchestrate a series of Amazon Athena queries that will run every day. Each query can run for more than 15 minutes. Which combination of steps will meet these requirements MOST cost-effectively? (Choose two.)

- A. Use an AWS Lambda function and the Athena Boto3 client start\_query\_execution API call to invoke the Athena queries programmatically.
- B. Create an AWS Step Functions workflow and add two state
- C. Add the first state before the Lambda function
- D. Configure the second state as a Wait state to periodically check whether the Athena query has finished using the Athena Boto3 get\_query\_execution API call
- E. Configure the workflow to invoke the next query when the current query has finished running.
- F. Use an AWS Glue Python shell job and the Athena Boto3 client start\_query\_execution API call to invoke the Athena queries programmatically.
- G. Use an AWS Glue Python shell script to run a sleep timer that checks every 5 minutes to determine whether the current Athena query has finished running successfully
- H. Configure the Python shell script to invoke the next query when the current query has finished running.
- I. Use Amazon Managed Workflows for Apache Airflow (Amazon MWAA) to orchestrate the Athena queries in AWS Batch.

**Answer:** AB

**Explanation:**

Option A and B are the correct answers because they meet the requirements most cost-effectively. Using an AWS Lambda function and the Athena Boto3 client start\_query\_execution API call to invoke the Athena queries programmatically is a simple and scalable way to orchestrate the queries. Creating an AWS Step Functions workflow and adding two states to check the query status and invoke the next query is a reliable and efficient way to handle the long-running queries. Option C is incorrect because using an AWS Glue Python shell job to invoke the Athena queries programmatically is more expensive than using a Lambda function, as it requires provisioning and running a Glue job for each query.

Option D is incorrect because using an AWS Glue Python shell script to run a sleep timer that checks every 5 minutes to determine whether the current Athena query has finished running successfully is not a cost-effective or reliable way to orchestrate the queries, as it wastes resources and time.

Option E is incorrect because using Amazon Managed Workflows for Apache Airflow (Amazon MWAA) to orchestrate the Athena queries in AWS Batch is an overkill solution that introduces unnecessary complexity and cost, as it requires setting up and managing an Airflow environment and an AWS Batch compute environment.

References:

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 5: Data Orchestration, Section 5.2: AWS Lambda, Section 5.3: AWS Step Functions, Pages 125-135

? Building Batch Data Analytics Solutions on AWS, Module 5: Data Orchestration, Lesson 5.1: AWS Lambda, Lesson 5.2: AWS Step Functions, Pages 1-15

? AWS Documentation Overview, AWS Lambda Developer Guide, Working with AWS Lambda Functions, Configuring Function Triggers, Using AWS Lambda with Amazon Athena, Pages 1-4

? AWS Documentation Overview, AWS Step Functions Developer Guide, Getting Started, Tutorial: Create a Hello World Workflow, Pages 1-8

**NEW QUESTION 10**

A financial company wants to implement a data mesh. The data mesh must support centralized data governance, data analysis, and data access control. The company has decided to use AWS Glue for data catalogs and extract, transform, and load (ETL) operations.

Which combination of AWS services will implement a data mesh? (Choose two.)

- A. Use Amazon Aurora for data storage
- B. Use an Amazon Redshift provisioned cluster for data analysis.
- C. Use Amazon S3 for data storage
- D. Use Amazon Athena for data analysis.
- E. Use AWS Glue DataBrew for centralized data governance and access control.
- F. Use Amazon RDS for data storage
- G. Use Amazon EMR for data analysis.
- H. Use AWS Lake Formation for centralized data governance and access control.

**Answer:** BE

**Explanation:**

A data mesh is an architectural framework that organizes data into domains and treats data as products that are owned and offered for consumption by different teams<sup>1</sup>. A data mesh requires a centralized layer for data governance and access control, as well as a distributed layer for data storage and analysis. AWS Glue can provide data catalogs and ETL operations for the data mesh, but it cannot provide data governance and access control by itself<sup>2</sup>. Therefore, the company

needs to use another AWS service for this purpose. AWS Lake Formation is a service that allows you to create, secure, and manage data lakes on AWS3. It integrates with AWS Glue and other AWS services to provide centralized data governance and access control for the data mesh. Therefore, option E is correct. For data storage and analysis, the company can choose from different AWS services depending on their needs and preferences. However, one of the benefits of a data mesh is that it enables data to be stored and processed in a decoupled and scalable way<sup>1</sup>. Therefore, using serverless or managed services that can handle large volumes and varieties of data is preferable. Amazon S3 is a highly scalable, durable, and secure object storage service that can store any type of data. Amazon Athena is a serverless interactive query service that can analyze data in Amazon S3 using standard SQL. Therefore, option B is a good choice for data storage and analysis in a data mesh. Option A, C, and D are not optimal because they either use relational databases that are not suitable for storing diverse and unstructured data, or they require more management and provisioning than serverless services. References:

? 1: What is a Data Mesh? - Data Mesh Architecture Explained - AWS

? 2: AWS Glue - Developer Guide

? 3: AWS Lake Formation - Features

? [4]: Design a data mesh architecture using AWS Lake Formation and AWS Glue

? [5]: Amazon S3 - Features

? [6]: Amazon Athena - Features

#### NEW QUESTION 10

A data engineer needs to build an extract, transform, and load (ETL) job. The ETL job will process daily incoming .csv files that users upload to an Amazon S3 bucket. The size of each S3 object is less than 100 MB.

Which solution will meet these requirements MOST cost-effectively?

- A. Write a custom Python applicatio
- B. Host the application on an Amazon Elastic Kubernetes Service (Amazon EKS) cluster.
- C. Write a PySpark ETL scrip
- D. Host the script on an Amazon EMR cluster.
- E. Write an AWS Glue PySpark jo
- F. Use Apache Spark to transform the data.
- G. Write an AWS Glue Python shell jo
- H. Use pandas to transform the data.

**Answer: D**

#### Explanation:

AWS Glue is a fully managed serverless ETL service that can handle various data sources and formats, including .csv files in Amazon S3. AWS Glue provides two types of jobs: PySpark and Python shell. PySpark jobs use Apache Spark to process large-scale data in parallel, while Python shell jobs use Python scripts to process small-scale data in a single execution environment. For this requirement, a Python shell job is more suitable and cost-effective, as the size of each S3 object is less than 100 MB, which does not require distributed processing. A Python shell job can use pandas, a popular Python library for data analysis, to transform the .csv data as needed. The other solutions are not optimal or relevant for this requirement. Writing a custom Python application and hosting it on an Amazon EKS cluster would require more effort and resources to set up and manage the Kubernetes environment, as well as to handle the data ingestion and transformation logic. Writing a PySpark ETL script and hosting it on an Amazon EMR cluster would also incur more costs and complexity to provision and configure the EMR cluster, as well as to use Apache Spark for processing small data files. Writing an AWS Glue PySpark job would also be less efficient and economical than a Python shell job, as it would involve unnecessary overhead and charges for using Apache Spark for small data files. References:

? AWS Glue

? Working with Python Shell Jobs

? pandas

? [AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide]

#### NEW QUESTION 11

A data engineer needs to create an AWS Lambda function that converts the format of data from .csv to Apache Parquet. The Lambda function must run only if a user uploads a .csv file to an Amazon S3 bucket.

Which solution will meet these requirements with the LEAST operational overhead?

- A. Create an S3 event notification that has an event type of s3:ObjectCreated:\*. Use a filter rule to generate notifications only when the suffix includes .cs
- B. Set the Amazon Resource Name (ARN) of the Lambda function as the destination for the event notification.
- C. Create an S3 event notification that has an event type of s3:ObjectTagging:\* for objects that have a tag set to .cs
- D. Set the Amazon Resource Name (ARN) of the Lambda function as the destination for the event notification.
- E. Create an S3 event notification that has an event type of s3:\*. Use a filter rule to generate notifications only when the suffix includes .cs
- F. Set the Amazon Resource Name (ARN) of the Lambda function as the destination for the event notification.
- G. Create an S3 event notification that has an event type of s3:ObjectCreated:\*. Use a filter rule to generate notifications only when the suffix includes .cs
- H. Set an Amazon Simple Notification Service (Amazon SNS) topic as the destination for the event notificatio
- I. Subscribe the Lambda function to the SNS topic.

**Answer: A**

#### Explanation:

Option A is the correct answer because it meets the requirements with the least operational overhead. Creating an S3 event notification that has an event type of s3:ObjectCreated:\* will trigger the Lambda function whenever a new object is created in the S3 bucket. Using a filter rule to generate notifications only when the suffix includes .csv will ensure that the Lambda function only runs for .csv files. Setting the ARN of the Lambda function as the destination for the event notification will directly invoke the Lambda function without any additional steps.

Option B is incorrect because it requires the user to tag the objects with .csv, which adds an extra step and increases the operational overhead.

Option C is incorrect because it uses an event type of s3:\*, which will trigger the Lambda function for any S3 event, not just object creation. This could result in unnecessary invocations and increased costs.

Option D is incorrect because it involves creating and subscribing to an SNS topic, which adds an extra layer of complexity and operational overhead.

References:

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 3: Data Ingestion and Transformation, Section 3.2: S3 Event Notifications and Lambda Functions, Pages 67-69

? Building Batch Data Analytics Solutions on AWS, Module 4: Data Transformation, Lesson 4.2: AWS Lambda, Pages 4-8

? AWS Documentation Overview, AWS Lambda Developer Guide, Working with AWS Lambda Functions, Configuring Function Triggers, Using AWS Lambda with Amazon S3, Pages 1-5

#### NEW QUESTION 14



A company extracts approximately 1 TB of data every day from data sources such as SAP HANA, Microsoft SQL Server, MongoDB, Apache Kafka, and Amazon DynamoDB. Some of the data sources have undefined data schemas or data schemas that change.

A data engineer must implement a solution that can detect the schema for these data sources. The solution must extract, transform, and load the data to an Amazon S3 bucket. The company has a service level agreement (SLA) to load the data into the S3 bucket within 15 minutes of data creation.

Which solution will meet these requirements with the LEAST operational overhead?

- A. Use Amazon EMR to detect the schema and to extract, transform, and load the data into the S3 bucket.
- B. Create a pipeline in Apache Spark.
- C. Use AWS Glue to detect the schema and to extract, transform, and load the data into the S3 bucket.
- D. Create a pipeline in Apache Spark.
- E. Create a PySpark program in AWS Lambda to extract, transform, and load the data into the S3 bucket.
- F. Create a stored procedure in Amazon Redshift to detect the schema and to extract, transform, and load the data into a Redshift Spectrum table.
- G. Access the table from Amazon S3.

**Answer: B**

**Explanation:**

AWS Glue is a fully managed service that provides a serverless data integration platform. It can automatically discover and categorize data from various sources, including SAP HANA, Microsoft SQL Server, MongoDB, Apache Kafka, and Amazon DynamoDB. It can also infer the schema of the data and store it in the AWS Glue Data Catalog, which is a central metadata repository. AWS Glue can then use the schema information to generate and run Apache Spark code to extract, transform, and load the data into an Amazon S3 bucket. AWS Glue can also monitor and optimize the performance and cost of the data pipeline, and handle any schema changes that may occur in the source data. AWS Glue can meet the SLA of loading the data into the S3 bucket within 15 minutes of data creation, as it can trigger the data pipeline based on events, schedules, or on-demand. AWS Glue has the least operational overhead among the options, as it does not require provisioning, configuring, or managing any servers or clusters. It also handles scaling, patching, and security automatically. References:

? AWS Glue

? [AWS Glue Data Catalog]

? [AWS Glue Developer Guide]

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

**NEW QUESTION 17**

A company receives a daily file that contains customer data in .xls format. The company stores the file in Amazon S3. The daily file is approximately 2 GB in size.

A data engineer concatenates the column in the file that contains customer first names and the column that contains customer last names. The data engineer needs to determine the number of distinct customers in the file.

Which solution will meet this requirement with the LEAST operational effort?

- A. Create and run an Apache Spark job in an AWS Glue notebook.
- B. Configure the job to read the S3 file and calculate the number of distinct customers.
- C. Create an AWS Glue crawler to create an AWS Glue Data Catalog of the S3 file.
- D. Run SQL queries from Amazon Athena to calculate the number of distinct customers.
- E. Create and run an Apache Spark job in Amazon EMR Serverless to calculate the number of distinct customers.
- F. Use AWS Glue DataBrew to create a recipe that uses the COUNT\_DISTINCT aggregate function to calculate the number of distinct customers.

**Answer: D**

**Explanation:**

AWS Glue DataBrew is a visual data preparation tool that allows you to clean, normalize, and transform data without writing code. You can use DataBrew to create recipes that define the steps to apply to your data, such as filtering, renaming, splitting, or aggregating columns. You can also use DataBrew to run jobs that execute the recipes on your data sources, such as Amazon S3, Amazon Redshift, or Amazon Aurora. DataBrew integrates with AWS Glue Data Catalog, which is a centralized metadata repository for your data assets<sup>1</sup>.

The solution that meets the requirement with the least operational effort is to use AWS Glue DataBrew to create a recipe that uses the COUNT\_DISTINCT aggregate function to calculate the number of distinct customers. This solution has the following advantages:

? It does not require you to write any code, as DataBrew provides a graphical user

interface that lets you explore, transform, and visualize your data. You can use DataBrew to concatenate the columns that contain customer first names and last names, and then use the COUNT\_DISTINCT aggregate function to count the number of unique values in the resulting column<sup>2</sup>.

? It does not require you to provision, manage, or scale any servers, clusters, or notebooks, as DataBrew is a fully managed service that handles all the infrastructure for you. DataBrew can automatically scale up or down the compute resources based on the size and complexity of your data and recipes<sup>1</sup>.

? It does not require you to create or update any AWS Glue Data Catalog entries, as

DataBrew can automatically create and register the data sources and targets in the Data Catalog. DataBrew can also use the existing Data Catalog entries to access the data in S3 or other sources<sup>3</sup>.

Option A is incorrect because it suggests creating and running an Apache Spark job in an AWS Glue notebook. This solution has the following disadvantages:

? It requires you to write code, as AWS Glue notebooks are interactive development environments that allow you to write, test, and debug Apache Spark code using Python or Scala. You need to use the Spark SQL or the Spark DataFrame API to read the S3 file and calculate the number of distinct customers.

? It requires you to provision and manage a development endpoint, which is a serverless Apache Spark environment that you can connect to your notebook. You need to specify the type and number of workers for your development endpoint, and monitor its status and metrics.

? It requires you to create or update the AWS Glue Data Catalog entries for the S3 file, either manually or using a crawler. You need to use the Data Catalog as a metadata store for your Spark job, and specify the database and table names in your code.

Option B is incorrect because it suggests creating an AWS Glue crawler to create an AWS Glue Data Catalog of the S3 file, and running SQL queries from Amazon Athena to calculate the number of distinct customers. This solution has the following disadvantages:

? It requires you to create and run a crawler, which is a program that connects to your data store, progresses through a prioritized list of classifiers to determine the schema for your data, and then creates metadata tables in the Data Catalog. You need to specify the data store, the IAM role, the schedule, and the output database for your crawler.

? It requires you to write SQL queries, as Amazon Athena is a serverless interactive query service that allows you to analyze data in S3 using standard SQL. You need to use Athena to concatenate the columns that contain customer first names and last names, and then use the COUNT(DISTINCT) aggregate function to count the number of unique values in the resulting column.

Option C is incorrect because it suggests creating and running an Apache Spark job in Amazon EMR Serverless to calculate the number of distinct customers. This solution has the following disadvantages:

? It requires you to write code, as Amazon EMR Serverless is a service that allows you to run Apache Spark jobs on AWS without provisioning or managing any infrastructure. You need to use the Spark SQL or the Spark DataFrame API to read the S3 file and calculate the number of distinct customers.

? It requires you to create and manage an Amazon EMR Serverless cluster, which is a fully managed and scalable Spark environment that runs on AWS Fargate. You need to specify the cluster name, the IAM role, the VPC, and the subnet for your cluster, and monitor its status and metrics.

? It requires you to create or update the AWS Glue Data Catalog entries for the S3 file, either manually or using a crawler. You need to use the Data Catalog as a metadata store for your Spark job, and specify the database and table names in your code.

**References:**

- ? 1: AWS Glue DataBrew - Features
- ? 2: Working with recipes - AWS Glue DataBrew
- ? 3: Working with data sources and data targets - AWS Glue DataBrew
- ? [4]: AWS Glue notebooks - AWS Glue
- ? [5]: Development endpoints - AWS Glue
- ? [6]: Populating the AWS Glue Data Catalog - AWS Glue
- ? [7]: Crawlers - AWS Glue
- ? [8]: Amazon Athena - Features
- ? [9]: Amazon EMR Serverless - Features
- ? [10]: Creating an Amazon EMR Serverless cluster - Amazon EMR
- ? [11]: Using the AWS Glue Data Catalog with Amazon EMR Serverless - Amazon EMR

**NEW QUESTION 19**

A company uses an on-premises Microsoft SQL Server database to store financial transaction data. The company migrates the transaction data from the on-premises database to AWS at the end of each month. The company has noticed that the cost to migrate data from the on-premises database to an Amazon RDS for SQL Server database has increased recently.

The company requires a cost-effective solution to migrate the data to AWS. The solution must cause minimal downtime for the applications that access the database.

Which AWS service should the company use to meet these requirements?

- A. AWS Lambda
- B. AWS Database Migration Service (AWS DMS)
- C. AWS Direct Connect
- D. AWS DataSync

**Answer: B**

**Explanation:**

AWS Database Migration Service (AWS DMS) is a cloud service that makes it possible to migrate relational databases, data warehouses, NoSQL databases, and other types of data stores to AWS quickly, securely, and with minimal downtime and zero data loss<sup>1</sup>. AWS DMS supports migration between 20-plus database and analytics engines, such as Microsoft SQL Server to Amazon RDS for SQL Server<sup>2</sup>. AWS DMS takes over many of the difficult or tedious tasks involved in a migration project, such as capacity analysis, hardware and software procurement, installation and administration, testing and debugging, and ongoing replication and monitoring<sup>1</sup>. AWS DMS is a cost-effective solution, as you only pay for the compute resources and additional log storage used during the migration process<sup>2</sup>. AWS DMS is the best solution for the company to migrate the financial transaction data from the on-premises Microsoft SQL Server database to AWS, as it meets the requirements of minimal downtime, zero data loss, and low cost.

Option A is not the best solution, as AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers, but it does not provide any built-in features for database migration. You would have to write your own code to extract, transform, and load the data from the source to the target, which would increase the operational overhead and complexity.

Option C is not the best solution, as AWS Direct Connect is a service that establishes a dedicated network connection from your premises to AWS, but it does not provide any built-in features for database migration. You would still need to use another service or tool to perform the actual data transfer, which would increase the cost and complexity.

Option D is not the best solution, as AWS DataSync is a service that makes it easy to transfer data between on-premises storage systems and AWS storage services, such as Amazon S3, Amazon EFS, and Amazon FSx for Windows File Server, but it does not support Amazon RDS for SQL Server as a target. You would have to use another service or tool to migrate the data from Amazon S3 to Amazon RDS for SQL Server, which would increase the latency and complexity.

**References:**

- ? Database Migration - AWS Database Migration Service - AWS
- ? What is AWS Database Migration Service?
- ? AWS Database Migration Service Documentation
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

**NEW QUESTION 22**

A media company wants to improve a system that recommends media content to customer based on user behavior and preferences. To improve the recommendation system, the company needs to incorporate insights from third-party datasets into the company's existing analytics platform.

The company wants to minimize the effort and time required to incorporate third-party datasets.

Which solution will meet these requirements with the LEAST operational overhead?

- A. Use API calls to access and integrate third-party datasets from AWS Data Exchange.
- B. Use API calls to access and integrate third-party datasets from AWS
- C. Use Amazon Kinesis Data Streams to access and integrate third-party datasets from AWS CodeCommit repositories.
- D. Use Amazon Kinesis Data Streams to access and integrate third-party datasets from Amazon Elastic Container Registry (Amazon ECR).

**Answer: A**

**Explanation:**

AWS Data Exchange is a service that makes it easy to find, subscribe to, and use third-party data in the cloud. It provides a secure and reliable way to access and integrate data from various sources, such as data providers, public datasets, or AWS services. Using AWS Data Exchange, you can browse and subscribe to data products that suit your needs, and then use API calls or the AWS Management Console to export the data to Amazon S3, where you can use it with your existing analytics platform. This solution minimizes the effort and time required to incorporate third-party datasets, as you do not need to set up and manage data pipelines, storage, or access controls. You also benefit from the data quality and freshness provided by the data providers, who can update their data products as frequently as needed<sup>12</sup>.

The other options are not optimal for the following reasons:

? B. Use API calls to access and integrate third-party datasets from AWS. This option is vague and does not specify which AWS service or feature is used to access and integrate third-party datasets. AWS offers a variety of services and features that can help with data ingestion, processing, and analysis, but not all of them are suitable for the given scenario. For example, AWS Glue is a serverless data integration service that can help you discover, prepare, and combine data from various sources, but it requires you to create and run data extraction, transformation, and loading (ETL) jobs, which can add operational overhead<sup>3</sup>.

? C. Use Amazon Kinesis Data Streams to access and integrate third-party datasets from AWS CodeCommit repositories. This option is not feasible, as AWS CodeCommit is a source control service that hosts secure Git-based repositories, not a data source that can be accessed by Amazon Kinesis Data Streams. Amazon Kinesis Data Streams is a service that enables you to capture, process, and analyze data streams in real time, such as clickstream data, application logs, or IoT telemetry. It does not support accessing and integrating data from AWS CodeCommit repositories, which are meant for storing and managing code, not data.



? D. Use Amazon Kinesis Data Streams to access and integrate third-party datasets from Amazon Elastic Container Registry (Amazon ECR). This option is also not feasible, as Amazon ECR is a fully managed container registry service that stores, manages, and deploys container images, not a data source that can be accessed by Amazon Kinesis Data Streams. Amazon Kinesis Data Streams does not support accessing and integrating data from Amazon ECR, which is meant for storing and managing container images, not data .

References:

- ? 1: AWS Data Exchange User Guide
- ? 2: AWS Data Exchange FAQs
- ? 3: AWS Glue Developer Guide
- ? : AWS CodeCommit User Guide
- ? : Amazon Kinesis Data Streams Developer Guide
- ? : Amazon Elastic Container Registry User Guide
- ? : Build a Continuous Delivery Pipeline for Your Container Images with Amazon ECR as Source

#### NEW QUESTION 25

A company has a production AWS account that runs company workloads. The company's security team created a security AWS account to store and analyze security logs from the production AWS account. The security logs in the production AWS account are stored in Amazon CloudWatch Logs.

The company needs to use Amazon Kinesis Data Streams to deliver the security logs to the security AWS account.

Which solution will meet these requirements?

- A. Create a destination data stream in the production AWS account
- B. In the security AWS account, create an IAM role that has cross-account permissions to Kinesis Data Streams in the production AWS account.
- C. Create a destination data stream in the security AWS account
- D. Create an IAM role and a trust policy to grant CloudWatch Logs the permission to put data into the stream
- E. Create a subscription filter in the security AWS account.
- F. Create a destination data stream in the production AWS account
- G. In the production AWS account, create an IAM role that has cross-account permissions to Kinesis Data Streams in the security AWS account.
- H. Create a destination data stream in the security AWS account
- I. Create an IAM role and a trust policy to grant CloudWatch Logs the permission to put data into the stream
- J. Create a subscription filter in the production AWS account.

**Answer: D**

#### Explanation:

Amazon Kinesis Data Streams is a service that enables you to collect, process, and analyze real-time streaming data. You can use Kinesis Data Streams to ingest data from various sources, such as Amazon CloudWatch Logs, and deliver it to different destinations, such as Amazon S3 or Amazon Redshift. To use Kinesis Data Streams to deliver the security logs from the production AWS account to the security AWS account, you need to create a destination data stream in the security AWS account. This data stream will receive the log data from the CloudWatch Logs service in the production AWS account. To enable this cross-account data delivery, you need to create an IAM role and a trust policy in the security AWS account. The IAM role defines the permissions that the CloudWatch Logs service needs to put data into the destination data stream. The trust policy allows the production AWS account to assume the IAM role. Finally, you need to create a subscription filter in the production AWS account. A subscription filter defines the pattern to match log events and the destination to send the matching events. In this case, the destination is the destination data stream in the security AWS account. This solution meets the requirements of using Kinesis Data Streams to deliver the security logs to the security AWS account. The other options are either not possible or not optimal. You cannot create a destination data stream in the production AWS account, as this would not deliver the data to the security AWS account. You cannot create a subscription filter in the security AWS account, as this would not capture the log events from the production AWS account. References:

? Using Amazon Kinesis Data Streams with Amazon CloudWatch Logs

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 3: Data Ingestion and Transformation, Section 3.3: Amazon Kinesis Data Streams

#### NEW QUESTION 29

An airline company is collecting metrics about flight activities for analytics. The company is conducting a proof of concept (POC) test to show how analytics can provide insights that the company can use to increase on-time departures.

The POC test uses objects in Amazon S3 that contain the metrics in .csv format. The POC test uses Amazon Athena to query the data. The data is partitioned in the S3 bucket by date.

As the amount of data increases, the company wants to optimize the storage solution to improve query performance.

Which combination of solutions will meet these requirements? (Choose two.)

- A. Add a randomized string to the beginning of the keys in Amazon S3 to get more throughput across partitions.
- B. Use an S3 bucket that is in the same account that uses Athena to query the data.
- C. Use an S3 bucket that is in the same AWS Region where the company runs Athena queries.
- D. Preprocess the .csv data to JSON format by fetching only the document keys that the query requires.
- E. Preprocess the .csv data to Apache Parquet format by fetching only the data blocks that are needed for predicates.

**Answer: CE**

#### Explanation:

Using an S3 bucket that is in the same AWS Region where the company runs Athena queries can improve query performance by reducing data transfer latency and costs. Preprocessing the .csv data to Apache Parquet format can also improve query performance by enabling columnar storage, compression, and partitioning, which can reduce the amount of data scanned and fetched by the query. These solutions can optimize the storage solution for the POC test without requiring much effort or changes to the existing data pipeline. The other solutions are not optimal or relevant for this requirement. Adding a randomized string to the beginning of the keys in Amazon S3 can improve the throughput across partitions, but it can also make the data harder to query and manage. Using an S3 bucket that is in the same account that uses Athena to query the data does not have any significant impact on query performance, as long as the proper permissions are granted. Preprocessing the .csv data to JSON format does not offer any benefits over the .csv format, as both are row-based and verbose formats that require more data scanning and fetching than columnar formats like Parquet. References:

? Best Practices When Using Athena with AWS Glue

? Optimizing Amazon S3 Performance

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

#### NEW QUESTION 34

A company stores daily records of the financial performance of investment portfolios in .csv format in an Amazon S3 bucket. A data engineer uses AWS Glue crawlers to crawl the S3 data.

The data engineer must make the S3 data accessible daily in the AWS Glue Data Catalog. Which solution will meet these requirements?

- A. Create an IAM role that includes the AmazonS3FullAccess policy
- B. Associate the role with the crawler
- C. Specify the S3 bucket path of the source data as the crawler's data store
- D. Create a daily schedule to run the crawler
- E. Configure the output destination to a new path in the existing S3 bucket.
- F. Create an IAM role that includes the AWSGlueServiceRole policy
- G. Associate the role with the crawler
- H. Specify the S3 bucket path of the source data as the crawler's data store
- I. Create a daily schedule to run the crawler
- J. Specify a database name for the output.
- K. Create an IAM role that includes the AmazonS3FullAccess policy
- L. Associate the role with the crawler
- M. Specify the S3 bucket path of the source data as the crawler's data store
- N. Allocate data processing units (DPUs) to run the crawler every day
- O. Specify a database name for the output.
- P. Create an IAM role that includes the AWSGlueServiceRole policy
- Q. Associate the role with the crawler
- R. Specify the S3 bucket path of the source data as the crawler's data store
- S. Allocate data processing units (DPUs) to run the crawler every day
- T. Configure the output destination to a new path in the existing S3 bucket.

**Answer:** B

**Explanation:**

To make the S3 data accessible daily in the AWS Glue Data Catalog, the data engineer needs to create a crawler that can crawl the S3 data and write the metadata to the Data Catalog. The crawler also needs to run on a daily schedule to keep the Data Catalog updated with the latest data. Therefore, the solution must include the following steps:

? Create an IAM role that has the necessary permissions to access the S3 data and

the Data Catalog. The AWSGlueServiceRole policy is a managed policy that grants these permissions<sup>1</sup>.

? Associate the role with the crawler.

? Specify the S3 bucket path of the source data as the crawler's data store. The crawler will scan the data and infer the schema and format<sup>2</sup>.

? Create a daily schedule to run the crawler. The crawler will run at the specified time every day and update the Data Catalog with any changes in the data<sup>3</sup>.

? Specify a database name for the output. The crawler will create or update a table in the Data Catalog under the specified database. The table will contain the metadata about the data in the S3 bucket, such as the location, schema, and classification.

Option B is the only solution that includes all these steps. Therefore, option B is the correct answer.

Option A is incorrect because it configures the output destination to a new path in the existing S3 bucket. This is unnecessary and may cause confusion, as the crawler does not write any data to the S3 bucket, only metadata to the Data Catalog.

Option C is incorrect because it allocates data processing units (DPUs) to run the crawler every day. This is also unnecessary, as DPUs are only used for AWS Glue ETL jobs, not crawlers.

Option D is incorrect because it combines the errors of option A and C. It configures the output destination to a new path in the existing S3 bucket and allocates DPUs to run the crawler every day, both of which are irrelevant for the crawler.

References:

? 1: AWS managed (predefined) policies for AWS Glue - AWS Glue

? 2: Data Catalog and crawlers in AWS Glue - AWS Glue

? 3: Scheduling an AWS Glue crawler - AWS Glue

? [4]: Parameters set on Data Catalog tables by crawler - AWS Glue

? [5]: AWS Glue pricing - Amazon Web Services (AWS)

**NEW QUESTION 36**

A company is developing an application that runs on Amazon EC2 instances. Currently, the data that the application generates is temporary. However, the company needs to persist the data, even if the EC2 instances are terminated.

A data engineer must launch new EC2 instances from an Amazon Machine Image (AMI) and configure the instances to preserve the data.

Which solution will meet this requirement?

- A. Launch new EC2 instances by using an AMI that is backed by an EC2 instance store volume that contains the application data
- B. Apply the default settings to the EC2 instances.
- C. Launch new EC2 instances by using an AMI that is backed by a root Amazon Elastic Block Store (Amazon EBS) volume that contains the application data
- D. Apply the default settings to the EC2 instances.
- E. Launch new EC2 instances by using an AMI that is backed by an EC2 instance store volume
- F. Attach an Amazon Elastic Block Store (Amazon EBS) volume to contain the application data
- G. Apply the default settings to the EC2 instances.
- H. Launch new EC2 instances by using an AMI that is backed by an Amazon Elastic Block Store (Amazon EBS) volume
- I. Attach an additional EC2 instance store volume to contain the application data
- J. Apply the default settings to the EC2 instances.

**Answer:** C

**Explanation:**

Amazon EC2 instances can use two types of storage volumes: instance store volumes and Amazon EBS volumes. Instance store volumes are ephemeral, meaning they are only attached to the instance for the duration of its life cycle. If the instance is stopped, terminated, or fails, the data on the instance store volume is lost. Amazon EBS volumes are persistent, meaning they can be detached from the instance and attached to another instance, and the data on the volume is preserved. To meet the requirement of persisting the data even if the EC2 instances are terminated, the data engineer must use Amazon EBS volumes to store the application data. The solution is to launch new EC2 instances by using an AMI that is backed by an EC2 instance store volume, which is the default option for most AMIs. Then, the data engineer must attach an Amazon EBS volume to each instance and configure the application to write the data to the EBS volume. This way, the data will be saved on the EBS volume and can be accessed by another instance if needed. The data engineer can apply the default settings to the EC2 instances, as there is no need to modify the instance type, security group, or IAM role for this solution. The other options are either not feasible or not optimal. Launching new EC2 instances by using an AMI that is backed by an EC2 instance store volume that contains the application data (option A) or by using an AMI that is backed by a root Amazon EBS volume that contains the application data (option B) would not work, as the data on the AMI would be outdated and overwritten by the new instances. Attaching an additional EC2 instance store volume to contain the application data (option D) would not work, as the data on the instance store volume would be lost if the instance is terminated. References:

- ? Amazon EC2 Instance Store
- ? Amazon EBS Volumes
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 2: Data Store Management, Section 2.1: Amazon EC2

**NEW QUESTION 39**

A company uses an Amazon Redshift provisioned cluster as its database. The Redshift cluster has five reserved ra3.4xlarge nodes and uses key distribution. A data engineer notices that one of the nodes frequently has a CPU load over 90%. SQL Queries that run on the node are queued. The other four nodes usually have a CPU load under 15% during daily operations. The data engineer wants to maintain the current number of compute nodes. The data engineer also wants to balance the load more evenly across all five compute nodes. Which solution will meet these requirements?

- A. Change the sort key to be the data column that is most often used in a WHERE clause of the SQL SELECT statement.
- B. Change the distribution key to the table column that has the largest dimension.
- C. Upgrade the reserved node from ra3.4xlarge to ra3.16xlarge.
- D. Change the primary key to be the data column that is most often used in a WHERE clause of the SQL SELECT statement.

**Answer: B**

**Explanation:**

Changing the distribution key to the table column that has the largest dimension will help to balance the load more evenly across all five compute nodes. The distribution key determines how the rows of a table are distributed among the slices of the cluster. If the distribution key is not chosen wisely, it can cause data skew, meaning some slices will have more data than others, resulting in uneven CPU load and query performance. By choosing the table column that has the largest dimension, meaning the column that has the most distinct values, as the distribution key, the data engineer can ensure that the rows are distributed more uniformly across the slices, reducing data skew and improving query performance.

The other options are not solutions that will meet the requirements. Option A, changing the sort key to be the data column that is most often used in a WHERE clause of the SQL SELECT statement, will not affect the data distribution or the CPU load. The sort key determines the order in which the rows of a table are stored on disk, which can improve the performance of range-restricted queries, but not the load balancing. Option C, upgrading the reserved node from ra3.4xlarge to ra3.16xlarge, will not maintain the current number of compute nodes, as it will increase the cost and the capacity of the cluster. Option D, changing the primary key to be the data column that is most often used in a WHERE clause of the SQL SELECT statement, will not affect the data distribution or the CPU load either. The primary key is a constraint that enforces the uniqueness of the rows in a table, but it does not influence the data layout or the query optimization. References:

- ? Choosing a data distribution style
- ? Choosing a data sort key
- ? Working with primary keys

**NEW QUESTION 43**

A company is building an analytics solution. The solution uses Amazon S3 for data lake storage and Amazon Redshift for a data warehouse. The company wants to use Amazon Redshift Spectrum to query the data that is in Amazon S3. Which actions will provide the FASTEST queries? (Choose two.)

- A. Use gzip compression to compress individual files to sizes that are between 1 GB and 5 GB.
- B. Use a columnar storage file format.
- C. Partition the data based on the most common query predicates.
- D. Split the data into files that are less than 10 KB.
- E. Use file formats that are not

**Answer: BC**

**Explanation:**

Amazon Redshift Spectrum is a feature that allows you to run SQL queries directly against data in Amazon S3, without loading or transforming the data. Redshift Spectrum can query various data formats, such as CSV, JSON, ORC, Avro, and Parquet. However, not all data formats are equally efficient for querying. Some data formats, such as CSV and JSON, are row-oriented, meaning that they store data as a sequence of records, each with the same fields. Row-oriented formats are suitable for loading and exporting data, but they are not optimal for analytical queries that often access only a subset of columns. Row-oriented formats also do not support compression or encoding techniques that can reduce the data size and improve the query performance.

On the other hand, some data formats, such as ORC and Parquet, are column-oriented, meaning that they store data as a collection of columns, each with a specific data type. Column-oriented formats are ideal for analytical queries that often filter, aggregate, or join data by columns. Column-oriented formats also support compression and encoding techniques that can reduce the data size and improve the query performance. For example, Parquet supports dictionary encoding, which replaces repeated values with numeric codes, and run-length encoding, which replaces consecutive identical values with a single value and a count. Parquet also supports various compression algorithms, such as Snappy, GZIP, and ZSTD, that can further reduce the data size and improve the query performance.

Therefore, using a columnar storage file format, such as Parquet, will provide faster queries, as it allows Redshift Spectrum to scan only the relevant columns and skip the rest, reducing the amount of data read from S3. Additionally, partitioning the data based on the most common query predicates, such as date, time, region, etc., will provide faster queries, as it allows Redshift Spectrum to prune the partitions that do not match the query criteria, reducing the amount of data scanned from S3. Partitioning also improves the performance of joins and aggregations, as it reduces data skew and shuffling.

The other options are not as effective as using a columnar storage file format and partitioning the data. Using gzip compression to compress individual files to sizes that are between 1 GB and 5 GB will reduce the data size, but it will not improve the query performance significantly, as gzip is not a splittable compression algorithm and requires decompression before reading. Splitting the data into files that are less than 10 KB will increase the number of files and the metadata overhead, which will degrade the query performance. Using file formats that are not supported by Redshift Spectrum, such as XML, will not work, as Redshift Spectrum will not be able to read or parse the data. References:

- ? Amazon Redshift Spectrum
- ? Choosing the Right Data Format
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 4: Data Lakes and Data Warehouses, Section 4.3: Amazon Redshift Spectrum

**NEW QUESTION 47**

A company stores details about transactions in an Amazon S3 bucket. The company wants to log all writes to the S3 bucket into another S3 bucket that is in the same AWS Region. Which solution will meet this requirement with the LEAST operational effort?

- A. Configure an S3 Event Notifications rule for all activities on the transactions S3 bucket to invoke an AWS Lambda function



- B. Program the Lambda function to write the event to Amazon Kinesis Data Firehose
- C. Configure Kinesis Data Firehose to write the event to the logs S3 bucket.
- D. Create a trail of management events in AWS CloudTrail
- E. Configure the trail to receive data from the transactions S3 bucket
- F. Specify an empty prefix and write-only event
- G. Specify the logs S3 bucket as the destination bucket.
- H. Configure an S3 Event Notifications rule for all activities on the transactions S3 bucket to invoke an AWS Lambda function
- I. Program the Lambda function to write the events to the logs S3 bucket.
- J. Create a trail of data events in AWS CloudTrail
- K. Configure the trail to receive data from the transactions S3 bucket
- L. Specify an empty prefix and write-only event
- M. Specify the logs S3 bucket as the destination bucket.

**Answer: D**

**Explanation:**

This solution meets the requirement of logging all writes to the S3 bucket into another S3 bucket with the least operational effort. AWS CloudTrail is a service that records the API calls made to AWS services, including Amazon S3. By creating a trail of data events, you can capture the details of the requests that are made to the transactions S3 bucket, such as the requester, the time, the IP address, and the response elements. By specifying an empty prefix and write-only events, you can filter the data events to only include the ones that write to the bucket. By specifying the logs S3 bucket as the destination bucket, you can store the CloudTrail logs in another S3 bucket that is in the same AWS Region. This solution does not require any additional coding or configuration, and it is more scalable and reliable than using S3 Event Notifications and Lambda functions. References:

- ? Logging Amazon S3 API calls using AWS CloudTrail
- ? Creating a trail for data events
- ? Enabling Amazon S3 server access logging

**NEW QUESTION 52**

A company is planning to use a provisioned Amazon EMR cluster that runs Apache Spark jobs to perform big data analysis. The company requires high reliability. A big data team must follow best practices for running cost-optimized and long-running workloads on Amazon EMR. The team must find a solution that will maintain the company's current level of performance.

Which combination of resources will meet these requirements MOST cost-effectively? (Choose two.)

- A. Use Hadoop Distributed File System (HDFS) as a persistent data store.
- B. Use Amazon S3 as a persistent data store.
- C. Use x86-based instances for core nodes and task nodes.
- D. Use Graviton instances for core nodes and task nodes.
- E. Use Spot Instances for all primary nodes.

**Answer: BD**

**Explanation:**

The best combination of resources to meet the requirements of high reliability, cost-optimization, and performance for running Apache Spark jobs on Amazon EMR is to use Amazon S3 as a persistent data store and Graviton instances for core nodes and task nodes.

Amazon S3 is a highly durable, scalable, and secure object storage service that can store any amount of data for a variety of use cases, including big data analytics<sup>1</sup>. Amazon S3 is a better choice than HDFS as a persistent data store for Amazon EMR, as it decouples the storage from the compute layer, allowing for more flexibility and cost-efficiency. Amazon S3 also supports data encryption, versioning, lifecycle management, and cross-region replication<sup>1</sup>. Amazon EMR integrates seamlessly with Amazon S3, using EMR File System (EMRFS) to access data stored in Amazon S3 buckets<sup>2</sup>. EMRFS also supports consistent view, which enables Amazon EMR to provide read-after-write consistency for Amazon S3 objects that are accessed through EMRFS<sup>2</sup>.

Graviton instances are powered by Arm-based AWS Graviton<sup>2</sup> processors that deliver up to 40% better price performance over comparable current generation x86-based instances<sup>3</sup>. Graviton instances are ideal for running workloads that are CPU-bound, memory-bound, or network-bound, such as big data analytics, web servers, and open-source databases<sup>3</sup>. Graviton instances are compatible with Amazon EMR, and can be used for both core nodes and task nodes. Core nodes are responsible for running the data processing frameworks, such as Apache Spark, and storing data in HDFS or the local file system. Task nodes are optional nodes that can be added to a cluster to increase the processing power and throughput. By using Graviton instances for both core nodes and task nodes, you can achieve higher performance and lower cost than using x86-based instances.

Using Spot Instances for all primary nodes is not a good option, as it can compromise the reliability and availability of the cluster. Spot Instances are spare EC2 instances that are available at up to 90% discount compared to On-Demand prices, but they can be interrupted by EC2 with a two-minute notice when EC2 needs the capacity back. Primary nodes are the nodes that run the cluster software, such as Hadoop, Spark, Hive, and Hue, and are essential for the cluster operation. If a primary node is interrupted by EC2, the cluster will fail or become unstable. Therefore, it is recommended to use On-Demand Instances or Reserved Instances for primary nodes, and use Spot Instances only for task nodes that can tolerate interruptions. References:

- ? Amazon S3 - Cloud Object Storage
- ? EMR File System (EMRFS)
- ? AWS Graviton2 Processor-Powered Amazon EC2 Instances
- ? [Plan and Configure EC2 Instances]
- ? [Amazon EC2 Spot Instances]
- ? [Best Practices for Amazon EMR]

**NEW QUESTION 56**

A company receives call logs as Amazon S3 objects that contain sensitive customer information. The company must protect the S3 objects by using encryption. The company must also use encryption keys that only specific employees can access.

Which solution will meet these requirements with the LEAST effort?

- A. Use an AWS CloudHSM cluster to store the encryption key
- B. Configure the process that writes to Amazon S3 to make calls to CloudHSM to encrypt and decrypt the object
- C. Deploy an IAM policy that restricts access to the CloudHSM cluster.
- D. Use server-side encryption with customer-provided keys (SSE-C) to encrypt the objects that contain customer information
- E. Restrict access to the keys that encrypt the objects.
- F. Use server-side encryption with AWS KMS keys (SSE-KMS) to encrypt the objects that contain customer information
- G. Configure an IAM policy that restricts access to the KMS keys that encrypt the objects.
- H. Use server-side encryption with Amazon S3 managed keys (SSE-S3) to encrypt the objects that contain customer information
- I. Configure an IAM policy that restricts access to the Amazon S3 managed keys that encrypt the objects.

**Answer:** C

**Explanation:**

Option C is the best solution to meet the requirements with the least effort because server-side encryption with AWS KMS keys (SSE-KMS) is a feature that allows you to encrypt data at rest in Amazon S3 using keys managed by AWS Key Management Service (AWS KMS). AWS KMS is a fully managed service that enables you to create and manage encryption keys for your AWS services and applications. AWS KMS also allows you to define granular access policies for your keys, such as who can use them to encrypt and decrypt data, and under what conditions. By using SSE-KMS, you can protect your S3 objects by using encryption keys that only specific employees can access, without having to manage the encryption and decryption process yourself.

Option A is not a good solution because it involves using AWS CloudHSM, which is a service that provides hardware security modules (HSMs) in the AWS Cloud. AWS CloudHSM allows you to generate and use your own encryption keys on dedicated hardware that is compliant with various standards and regulations.

However, AWS CloudHSM is not a fully managed service and requires more effort to set up and maintain than AWS KMS. Moreover, AWS CloudHSM does not integrate with Amazon S3, so you have to configure the process that writes to S3 to make calls to CloudHSM to encrypt and decrypt the objects, which adds complexity and latency to the data protection process. Option B is not a good solution because it involves using server-side encryption with customer-provided keys (SSE-C), which is a feature that allows you to encrypt data at rest in Amazon S3 using keys that you provide and manage yourself. SSE-C requires you to send your encryption key along with each request to upload or retrieve an object. However, SSE-C does not provide any mechanism to restrict access to the keys that encrypt the objects, so you have to implement your own key management and access control system, which adds more effort and risk to the data protection process.

Option D is not a good solution because it involves using server-side encryption with Amazon S3 managed keys (SSE-S3), which is a feature that allows you to encrypt data at rest in Amazon S3 using keys that are managed by Amazon S3. SSE-S3 automatically encrypts and decrypts your objects as they are uploaded and downloaded from S3. However, SSE-S3 does not allow you to control who can access the encryption keys or under what conditions. SSE-S3 uses a single encryption key for each S3 bucket, which is shared by all users who have access to the bucket. This means that you cannot restrict access to the keys that encrypt the objects by specific employees, which does not meet the requirements.

References:

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

? Protecting Data Using Server-Side Encryption with AWS KMS–Managed Encryption Keys (SSE-KMS) - Amazon Simple Storage Service

? What is AWS Key Management Service? - AWS Key Management Service

? What is AWS CloudHSM? - AWS CloudHSM

? Protecting Data Using Server-Side Encryption with Customer-Provided Encryption Keys (SSE-C) - Amazon Simple Storage Service

? Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3) - Amazon Simple Storage Service

**NEW QUESTION 59**

.....

## Thank You for Trying Our Product

### We offer two products:

1st - We have Practice Tests Software with Actual Exam Questions

2nd - Questions and Answers in PDF Format

### AWS-Certified-Data-Engineer-Associate Practice Exam Features:

- \* AWS-Certified-Data-Engineer-Associate Questions and Answers Updated Frequently
- \* AWS-Certified-Data-Engineer-Associate Practice Questions Verified by Expert Senior Certified Staff
- \* AWS-Certified-Data-Engineer-Associate Most Realistic Questions that Guarantee you a Pass on Your FirstTry
- \* AWS-Certified-Data-Engineer-Associate Practice Test Questions in Multiple Choice Formats and Updatesfor 1 Year

**100% Actual & Verified — Instant Download, Please Click**  
**[Order The AWS-Certified-Data-Engineer-Associate Practice Test Here](#)**