

Amazon

Exam Questions AWS-Certified-Data-Engineer-Associate

AWS Certified Data Engineer - Associate (DEA-C01)



NEW QUESTION 1

A company uses Amazon Athena for one-time queries against data that is in Amazon S3. The company has several use cases. The company must implement permission controls to separate query processes and access to query history among users, teams, and applications that are in the same AWS account. Which solution will meet these requirements?

- A. Create an S3 bucket for each use cas
- B. Create an S3 bucket policy that grants permissions to appropriate individual IAM user
- C. Apply the S3 bucket policy to the S3 bucket.
- D. Create an Athena workgroup for each use cas
- E. Apply tags to the workgrou
- F. Create an IAM policy that uses the tags to apply appropriate permissions to the workgroup.
- G. Create an IAM role for each use cas
- H. Assign appropriate permissions to the role for each use cas
- I. Associate the role with Athena.
- J. Create an AWS Glue Data Catalog resource policy that grants permissions to appropriate individual IAM users for each use cas
- K. Apply the resource policy to the specific tables that Athena uses.

Answer: B

Explanation:

Athena workgroups are a way to isolate query execution and query history among users, teams, and applications that share the same AWS account. By creating a workgroup for each use case, the company can control the access and actions on the workgroup resource using resource-level IAM permissions or identity-based IAM policies. The company can also use tags to organize and identify the workgroups, and use them as conditions in the IAM policies to grant or deny permissions to the workgroup. This solution meets the requirements of separating query processes and access to query history among users, teams, and applications that are in the same AWS account. References:

- ? Athena Workgroups
- ? IAM policies for accessing workgroups
- ? Workgroup example policies

NEW QUESTION 2

A company stores datasets in JSON format and .csv format in an Amazon S3 bucket. The company has Amazon RDS for Microsoft SQL Server databases, Amazon DynamoDB tables that are in provisionedcapacity mode, and an Amazon Redshift cluster. A data engineering team must develop a solution that will give data scientists the ability to query all data sources by using syntax similar to SQL.

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

- A. Use AWS Glue to crawl the data source
- B. Store metadata in the AWS Glue Data Catalog
- C. Use Amazon Athena to query the data
- D. Use SQL for structured data source
- E. Use PartiQL for data that is stored in JSON format.
- F. Use AWS Glue to crawl the data source
- G. Store metadata in the AWS Glue Data Catalog
- H. Use Redshift Spectrum to query the data
- I. Use SQL for structured data source
- J. Use PartiQL for data that is stored in JSON format.
- K. Use AWS Glue to crawl the data source
- L. Store metadata in the AWS Glue Data Catalog
- M. Use AWS Glue jobs to transform data that is in JSON format to Apache Parquet or .csv format
- N. Store the transformed data in an S3 bucket
- O. Use Amazon Athena to query the original and transformed data from the S3 bucket.
- P. Use AWS Lake Formation to create a data lake
- Q. Use Lake Formation jobs to transform the data from all data sources to Apache Parquet format
- R. Store the transformed data in an S3 bucket
- S. Use Amazon Athena or Redshift Spectrum to query the data.

Answer: A

Explanation:

The best solution to meet the requirements of giving data scientists the ability to query all data sources by using syntax similar to SQL with the least operational overhead is to use AWS Glue to crawl the data sources, store metadata in the AWS Glue Data Catalog, use Amazon Athena to query the data, use SQL for structured data sources, and use PartiQL for data that is stored in JSON format.

AWS Glue is a serverless data integration service that makes it easy to prepare, clean, enrich, and move data between data stores¹. 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 Catalog². The Data Catalog is a persistent metadata store that contains table definitions, job definitions, and other control information to help you manage your AWS Glue components³. You can use AWS Glue to crawl the data sources, such as Amazon S3, Amazon RDS for Microsoft SQL Server, and Amazon DynamoDB, and store the metadata in the Data Catalog.

Amazon Athena is a serverless, interactive query service that makes it easy to analyze data directly in Amazon S3 using standard SQL or Python⁴. Amazon Athena also supports PartiQL, a SQL-compatible query language that lets you query, insert, update, and delete data from semi-structured and nested data, such as JSON. You can use Amazon Athena to query the data from the Data Catalog using SQL for structured data sources, such as .csv files and relational databases, and PartiQL for data that is stored in JSON format. You can also use Athena to query data from other data sources, such as Amazon Redshift, using federated queries.

Using AWS Glue and Amazon Athena to query all data sources by using syntax similar to SQL is the least operational overhead solution, as you do not need to provision, manage, or scale any infrastructure, and you pay only for the resources you use. AWS Glue charges you based on the compute time and the data processed by your crawlers and ETL jobs¹. Amazon Athena charges you based on the amount of data scanned by your queries. You can also reduce the cost and improve the performance of your queries by using compression, partitioning, and columnar formats for your data in Amazon S3.

Option B is not the best solution, as using AWS Glue to crawl the data sources, store metadata in the AWS Glue Data Catalog, and use Redshift Spectrum to query the data, would incur more costs and complexity than using Amazon Athena. Redshift Spectrum is a feature of Amazon Redshift, a fully managed data warehouse service, that allows you to query and join data across your data warehouse and your data lake using standard SQL. While Redshift Spectrum is powerful and useful for many data warehousing scenarios, it is not necessary or cost-effective for querying all data sources by using syntax similar to SQL. Redshift Spectrum charges you based on the amount of data scanned by your queries, which is similar to Amazon Athena, but it also requires you to have an

Amazon Redshift cluster, which charges you based on the node type, the number of nodes, and the duration of the cluster⁵. These costs can add up quickly, especially if you have large volumes of data and complex queries. Moreover, using Redshift Spectrum would introduce additional latency and complexity, as you would have to provision and manage the cluster, and create an external schema and database for the data in the Data Catalog, instead of querying it directly from Amazon Athena.

Option C is not the best solution, as using AWS Glue to crawl the data sources, store metadata in the AWS Glue Data Catalog, use AWS Glue jobs to transform data that is in JSON format to Apache Parquet or .csv format, store the transformed data in an S3 bucket, and use Amazon Athena to query the original and transformed data from the S3 bucket, would incur more costs and complexity than using Amazon Athena with PartiQL. AWS Glue jobs are ETL scripts that you can write in Python or Scala to transform your data and load it to your target data store. Apache Parquet is a columnar storage format that can improve the performance of analytical queries by reducing the amount of data that needs to be scanned and providing efficient compression and encoding schemes⁶. While using AWS Glue jobs and Parquet can improve the performance and reduce the cost of your queries, they would also increase the complexity and the operational overhead of the data pipeline, as you would have to write, run, and monitor the ETL jobs, and store the transformed data in a separate location in Amazon S3. Moreover, using AWS Glue jobs and Parquet would introduce additional latency, as you would have to wait for the ETL jobs to finish before querying the transformed data.

Option D is not the best solution, as using AWS Lake Formation to create a data lake, use Lake Formation jobs to transform the data from all data sources to Apache Parquet format, store the transformed data in an S3 bucket, and use Amazon Athena or Redshift Spectrum to query the data, would incur more costs and complexity than using Amazon Athena with PartiQL. AWS Lake Formation is a service that helps you centrally govern, secure, and globally share data for analytics and machine learning⁷. Lake Formation jobs are ETL jobs that you can create and run using the Lake Formation console or API. While using Lake Formation and Parquet can improve the performance and reduce the cost of your queries, they would also increase the complexity and the operational overhead of the data pipeline, as you would have to create, run, and monitor the Lake Formation jobs, and store the transformed data in a separate location in Amazon S3. Moreover, using Lake Formation and Parquet would introduce additional latency, as you would have to wait for the Lake Formation jobs to finish before querying the transformed data. Furthermore, using Redshift Spectrum to query the data would also incur the same costs and complexity as mentioned in option B. References:

- ? What is Amazon Athena?
- ? Data Catalog and crawlers in AWS Glue
- ? AWS Glue Data Catalog
- ? Columnar Storage Formats
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide
- ? AWS Glue Schema Registry
- ? What is AWS Glue?
- ? Amazon Redshift Serverless
- ? Amazon Redshift provisioned clusters
- ? [Querying external data using Amazon Redshift Spectrum]
- ? [Using stored procedures in Amazon Redshift]
- ? [What is AWS Lambda?]
- ? [PartiQL for Amazon Athena]
- ? [Federated queries in Amazon Athena]
- ? [Amazon Athena pricing]
- ? [Top 10 performance tuning tips for Amazon Athena]
- ? [AWS Glue ETL jobs]
- ? [AWS Lake Formation jobs]

NEW QUESTION 3

A data engineer is configuring Amazon SageMaker Studio to use AWS Glue interactive sessions to prepare data for machine learning (ML) models. The data engineer receives an access denied error when the data engineer tries to prepare the data by using SageMaker Studio. Which change should the engineer make to gain access to SageMaker Studio?

- A. Add the AWSGlueServiceRole managed policy to the data engineer's IAM user.
- B. Add a policy to the data engineer's IAM user that includes the sts:AssumeRole action for the AWS Glue and SageMaker service principals in the trust policy.
- C. Add the AmazonSageMakerFullAccess managed policy to the data engineer's IAM user.
- D. Add a policy to the data engineer's IAM user that allows the sts:AddAssociation action for the AWS Glue and SageMaker service principals in the trust policy.

Answer: B

Explanation:

This solution meets the requirement of gaining access to SageMaker Studio to use AWS Glue interactive sessions. AWS Glue interactive sessions are a way to use AWS Glue DataBrew and AWS Glue Data Catalog from within SageMaker Studio. To use AWS Glue interactive sessions, the data engineer's IAM user needs to have permissions to assume the AWS Glue service role and the SageMaker execution role. By adding a policy to the data engineer's IAM user that includes the sts:AssumeRole action for the AWS Glue and SageMaker service principals in the trust policy, the data engineer can grant these permissions and avoid the access denied error. The other options are not sufficient or necessary to resolve the error. References:

- ? Get started with data integration from Amazon S3 to Amazon Redshift using AWS Glue interactive sessions
- ? Troubleshoot Errors - Amazon SageMaker
- ? AccessDeniedException on sagemaker:CreateDomain in AWS SageMaker Studio, despite having SageMakerFullAccess

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 company uses Amazon S3 to store semi-structured data in a transactional data lake. Some of the data files are small, but other data files are tens of terabytes. A data engineer must perform a change data capture (CDC) operation to identify changed data from the data source. The data source sends a full snapshot as a JSON file every day and ingests the changed data into the data lake.

Which solution will capture the changed data MOST cost-effectively?

- A. Create an AWS Lambda function to identify the changes between the previous data and the current data.
- B. Configure the Lambda function to ingest the changes into the data lake.
- C. Ingest the data into Amazon RDS for MySQL.
- D. Use AWS Database Migration Service (AWS DMS) to write the changed data to the data lake.
- E. Use an open source data lake format to merge the data source with the S3 data lake to insert the new data and update the existing data.
- F. Ingest the data into an Amazon Aurora MySQL DB instance that runs Aurora Serverless.
- G. Use AWS Database Migration Service (AWS DMS) to write the changed data to the data lake.

Answer: C

Explanation:

An open source data lake format, such as Apache Parquet, Apache ORC, or Delta Lake, is a cost-effective way to perform a change data capture (CDC) operation on semi-structured data stored in Amazon S3. An open source data lake format allows you to query data directly from S3 using standard SQL, without the need to move or copy data to another service. An open source data lake format also supports schema evolution, meaning it can handle changes in the data structure over time. An open source data lake format also supports upserts, meaning it can insert new data and update existing data in the same operation, using a merge command. This way, you can efficiently capture the changes from the data source and apply them to the S3 data lake, without duplicating or losing any data. The other options are not as cost-effective as using an open source data lake format, as they involve additional steps or costs. Option A requires you to create and maintain an AWS Lambda function, which can be complex and error-prone. AWS Lambda also has some limits on the execution time, memory, and concurrency, which can affect the performance and reliability of the CDC operation. Option B and D require you to ingest the data into a relational database service, such as Amazon RDS or Amazon Aurora, which can be expensive and unnecessary for semi-structured data. AWS Database Migration Service (AWS DMS) can write the changed data to the data lake, but it also charges you for the data replication and transfer. Additionally, AWS DMS does not support JSON as a source data type, so you would need to convert the data to a supported format before using AWS DMS. References:

- ? What is a data lake?
- ? Choosing a data format for your data lake
- ? Using the MERGE INTO command in Delta Lake
- ? [AWS Lambda quotas]
- ? [AWS Database Migration Service quotas]

NEW QUESTION 6

A company is planning to upgrade its Amazon Elastic Block Store (Amazon EBS) General Purpose SSD storage from gp2 to gp3. The company wants to prevent any interruptions in its Amazon EC2 instances that will cause data loss during the migration to the upgraded storage.

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

- A. Create snapshots of the gp2 volume.
- B. Create new gp3 volumes from the snapshot.
- C. Attach the new gp3 volumes to the EC2 instances.
- D. Create new gp3 volume.
- E. Gradually transfer the data to the new gp3 volume.
- F. When the transfer is complete, mount the new gp3 volumes to the EC2 instances to replace the gp2 volumes.
- G. Change the volume type of the existing gp2 volumes to gp3. Enter new values for volume size, IOPS, and throughput.
- H. Use AWS DataSync to create new gp3 volume.
- I. Transfer the data from the original gp2 volumes to the new gp3 volumes.

Answer: C

Explanation:

Changing the volume type of the existing gp2 volumes to gp3 is the easiest and fastest way to migrate to the new storage type without any downtime or data loss. You can use the AWS Management Console, the AWS CLI, or the Amazon EC2 API to modify the volume type, size, IOPS, and throughput of your gp2 volumes. The modification takes effect immediately, and you can monitor the progress of the modification using CloudWatch. The other options are either more complex or require additional steps, such as creating snapshots, transferring data, or attaching new volumes, which can increase the operational overhead and the risk of errors. References:

- ? Migrating Amazon EBS volumes from gp2 to gp3 and save up to 20% on costs (Section: How to migrate from gp2 to gp3)
- ? Switching from gp2 Volumes to gp3 Volumes to Lower AWS EBS Costs (Section: How to Switch from GP2 Volumes to GP3 Volumes)
- ? Modifying the volume type, IOPS, or size of an EBS volume - Amazon Elastic Compute Cloud (Section: Modifying the volume type)

NEW QUESTION 7

A company receives .csv files that contain physical address data. The data is in columns that have the following names: Door_No, Street_Name, City, and Zip_Code. The company wants to create a single column to store these values in the following format:

```
{
  "Door_No": "24",
  "Street_Name": "AAA street",
  "City": "BBB",
  "Zip_Code": "111111"
}
```

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

- A. Use AWS Glue DataBrew to read the file
- B. Use the NEST TO ARRAY transformation to create the new column.
- C. Use AWS Glue DataBrew to read the file
- D. Use the NEST TO MAP transformation to create the new column.
- E. Use AWS Glue DataBrew to read the file
- F. Use the PIVOT transformation to create the new column.
- G. Write a Lambda function in Python to read the file
- H. Use the Python data dictionary type to create the new column.

Answer: B

Explanation:

The NEST TO MAP transformation allows you to combine multiple columns into a single column that contains a JSON object with key-value pairs. This is the easiest way to achieve the desired format for the physical address data, as you can simply select the columns to nest and specify the keys for each column. The NEST TO ARRAY transformation creates a single column that contains an array of values, which is not the same as the JSON object format. The PIVOT transformation reshapes the data by creating new columns from unique values in a selected column, which is not applicable for this use case. Writing a Lambda function in Python requires more coding effort than using AWS Glue DataBrew, which provides a visual and interactive interface for data transformations.

References:

? 7 most common data preparation transformations in AWS Glue DataBrew (Section: Nesting and unnesting columns)

? NEST TO MAP - AWS Glue DataBrew (Section: Syntax)

NEW QUESTION 8

A manufacturing company wants to collect data from sensors. A data engineer needs to implement a solution that ingests sensor data in near real time. The solution must store the data to a persistent data store. The solution must store the data in nested JSON format. The company must have the ability to query from the data store with a latency of less than 10 milliseconds.

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

- A. Use a self-hosted Apache Kafka cluster to capture the sensor data
- B. Store the data in Amazon S3 for querying.
- C. Use AWS Lambda to process the sensor data
- D. Store the data in Amazon S3 for querying.
- E. Use Amazon Kinesis Data Streams to capture the sensor data
- F. Store the data in Amazon DynamoDB for querying.
- G. Use Amazon Simple Queue Service (Amazon SQS) to buffer incoming sensor data
- H. Use AWS Glue to store the data in Amazon RDS for querying.

Answer: C

Explanation:

Amazon Kinesis Data Streams is a service that enables you to collect, process, and analyze streaming data in real time. You can use Kinesis Data Streams to capture sensor data from various sources, such as IoT devices, web applications, or mobile apps. You can create data streams that can scale up to handle any amount of data from thousands of producers. You can also use the Kinesis Client Library (KCL) or the Kinesis Data Streams API to write applications that process and analyze the data in the streams¹. Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. You can use DynamoDB to store the sensor data in nested JSON format, as DynamoDB supports document data types, such as lists and maps. You can also use DynamoDB to query the data with a latency of less than 10 milliseconds, as DynamoDB offers single-digit millisecond performance for any scale of data. You can use the DynamoDB API or the AWS SDKs to perform queries on the data, such as using key-value lookups, scans, or queries².

The solution that meets the requirements with the least operational overhead is to use Amazon Kinesis Data Streams to capture the sensor data and store the data in Amazon DynamoDB for querying. This solution has the following advantages:

? It does not require you to provision, manage, or scale any servers, clusters, or queues, as Kinesis Data Streams and DynamoDB are fully managed services that handle all the infrastructure for you. This reduces the operational complexity and cost of running your solution.

? It allows you to ingest sensor data in near real time, as Kinesis Data Streams can capture data records as they are produced and deliver them to your applications within seconds. You can also use Kinesis Data Firehose to load the data from the streams to DynamoDB automatically and continuously³.

? It allows you to store the data in nested JSON format, as DynamoDB supports document data types, such as lists and maps. You can also use DynamoDB Streams to capture changes in the data and trigger actions, such as sending notifications or updating other databases.

? It allows you to query the data with a latency of less than 10 milliseconds, as DynamoDB offers single-digit millisecond performance for any scale of data. You can also use DynamoDB Accelerator (DAX) to improve the read performance by caching frequently accessed data.

Option A is incorrect because it suggests using a self-hosted Apache Kafka cluster to capture the sensor data and store the data in Amazon S3 for querying. This solution has the following disadvantages:

? It requires you to provision, manage, and scale your own Kafka cluster, either on EC2 instances or on-premises servers. This increases the operational complexity and cost of running your solution.

? It does not allow you to query the data with a latency of less than 10 milliseconds, as Amazon S3 is an object storage service that is not optimized for low-latency queries. You need to use another service, such as Amazon Athena or Amazon Redshift Spectrum, to query the data in S3, which may incur additional costs and latency.

Option B is incorrect because it suggests using AWS Lambda to process the sensor data and store the data in Amazon S3 for querying. This solution has the following disadvantages:

? It does not allow you to ingest sensor data in near real time, as Lambda is a serverless compute service that runs code in response to events. You need to use another service, such as API Gateway or Kinesis Data Streams, to trigger Lambda functions with sensor data, which may add extra latency and complexity to your solution.

? It does not allow you to query the data with a latency of less than 10 milliseconds, as Amazon S3 is an object storage service that is not optimized for low-latency queries. You need to use another service, such as Amazon Athena or Amazon Redshift Spectrum, to query the data in S3, which may incur additional costs and latency.

Option D is incorrect because it suggests using Amazon Simple Queue Service (Amazon SQS) to buffer incoming sensor data and use AWS Glue to store the data in Amazon RDS for querying. This solution has the following disadvantages:

? It does not allow you to ingest sensor data in near real time, as Amazon SQS is a message queue service that delivers messages in a best-effort manner. You need to use another service, such as Lambda or EC2, to poll the messages from the queue and process them, which may add extra latency and complexity to your solution.

? It does not allow you to store the data in nested JSON format, as Amazon RDS is a relational database service that supports structured data types, such as tables and columns. You need to use another service, such as AWS Glue, to transform the data from JSON to relational format, which may add extra cost and overhead to your solution.

References:

- ? 1: Amazon Kinesis Data Streams - Features
- ? 2: Amazon DynamoDB - Features
- ? 3: Loading Streaming Data into Amazon DynamoDB - Amazon Kinesis Data Firehose
- ? [4]: Capturing Table Activity with DynamoDB Streams - Amazon DynamoDB
- ? [5]: Amazon DynamoDB Accelerator (DAX) - Features
- ? [6]: Amazon S3 - Features
- ? [7]: AWS Lambda - Features
- ? [8]: Amazon Simple Queue Service - Features
- ? [9]: Amazon Relational Database Service - Features
- ? [10]: Working with JSON in Amazon RDS - Amazon Relational Database Service
- ? [11]: AWS Glue - Features

NEW QUESTION 9

A data engineer must orchestrate a data pipeline that consists of one AWS Lambda function and one AWS Glue job. The solution must integrate with AWS services.

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

- A. Use an AWS Step Functions workflow that includes a state machine
- B. Configure the state machine to run the Lambda function and then the AWS Glue job.
- C. Use an Apache Airflow workflow that is deployed on an Amazon EC2 instance
- D. Define a directed acyclic graph (DAG) in which the first task is to call the Lambda function and the second task is to call the AWS Glue job.
- E. Use an AWS Glue workflow to run the Lambda function and then the AWS Glue job.
- F. Use an Apache Airflow workflow that is deployed on Amazon Elastic Kubernetes Service (Amazon EKS). Define a directed acyclic graph (DAG) in which the first task is to call the Lambda function and the second task is to call the AWS Glue job.

Answer: A

Explanation:

AWS Step Functions is a service that allows you to coordinate multiple AWS services into serverless workflows. You can use Step Functions to create state machines that define the sequence and logic of the tasks in your workflow. Step Functions supports various types of tasks, such as Lambda functions, AWS Glue jobs, Amazon EMR clusters, Amazon ECS tasks, etc. You can use Step Functions to monitor and troubleshoot your workflows, as well as to handle errors and retries.

Using an AWS Step Functions workflow that includes a state machine to run the Lambda function and then the AWS Glue job will meet the requirements with the least management overhead, as it leverages the serverless and managed capabilities of Step Functions. You do not need to write any code to orchestrate the tasks in your workflow, as you can use the Step Functions console or the AWS Serverless Application Model (AWS SAM) to define and deploy your state machine. You also do not need to provision or manage any servers or clusters, as Step Functions scales automatically based on the demand.

The other options are not as efficient as using an AWS Step Functions workflow. Using an Apache Airflow workflow that is deployed on an Amazon EC2 instance or on Amazon Elastic Kubernetes Service (Amazon EKS) will require more management overhead, as you will need to provision, configure, and maintain the EC2 instance or the EKS cluster, as well as the Airflow components. You will also need to write and maintain the Airflow DAGs to orchestrate the tasks in your workflow. Using an AWS Glue workflow to run the Lambda function and then the AWS Glue job will not work, as AWS Glue workflows only support AWS Glue jobs and crawlers as tasks, not Lambda functions. References:

- ? AWS Step Functions
- ? AWS Glue
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 6: Data Integration and Transformation, Section 6.3: AWS Step Functions

NEW QUESTION 10

During a security review, a company identified a vulnerability in an AWS Glue job. The company discovered that credentials to access an Amazon Redshift cluster were hard coded in the job script.

A data engineer must remediate the security vulnerability in the AWS Glue job. The solution must securely store the credentials.

Which combination of steps should the data engineer take to meet these requirements? (Choose two.)

- A. Store the credentials in the AWS Glue job parameters.
- B. Store the credentials in a configuration file that is in an Amazon S3 bucket.
- C. Access the credentials from a configuration file that is in an Amazon S3 bucket by using the AWS Glue job.
- D. Store the credentials in AWS Secrets Manager.
- E. Grant the AWS Glue job 1AM role access to the stored credentials.

Answer: DE

Explanation:

AWS Secrets Manager is a service that allows you to securely store and manage secrets, such as database credentials, API keys, passwords, etc. You can use Secrets Manager to encrypt, rotate, and audit your secrets, as well as to control access to them using fine-grained policies. AWS Glue is a fully managed service that provides a serverless data integration platform for data preparation, data cataloging, and data loading. AWS Glue jobs allow you to transform and load data from various sources into various targets, using either a graphical interface (AWS Glue Studio) or a code-based interface (AWS Glue console or AWS Glue API). Storing the credentials in AWS Secrets Manager and granting the AWS Glue job 1AM role access to the stored credentials will meet the requirements, as it will remediate the security vulnerability in the AWS Glue job and securely store the credentials. By using AWS Secrets Manager, you can avoid hard coding the credentials in the job script, which is a bad practice that exposes the credentials to unauthorized access or leakage. Instead, you can store the credentials as a secret in Secrets Manager and reference the secret name or ARN in the job script. You can also use Secrets Manager to encrypt the credentials using AWS Key Management Service (AWS KMS), rotate the credentials automatically or on demand, and monitor the access to the credentials using AWS CloudTrail. By granting

the AWS Glue job 1AM role access to the stored credentials, you can use the principle of least privilege to ensure that only the AWS Glue job can retrieve the credentials from Secrets Manager. You can also use resource-based or tag-based policies to further restrict the access to the credentials. The other options are not as secure as storing the credentials in AWS Secrets Manager and granting the AWS Glue job 1AM role access to the stored credentials. Storing the credentials in the AWS Glue job parameters will not remediate the security vulnerability, as the job parameters are still visible in the AWS Glue console and API. Storing the credentials in a configuration file that is in an Amazon S3 bucket and accessing the credentials from the configuration file by using the AWS Glue job will not be as secure as using Secrets Manager, as the configuration file may not be encrypted or rotated, and the access to the file may not be audited or controlled. References:
 ? AWS Secrets Manager
 ? AWS Glue
 ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 6: Data Integration and Transformation, Section 6.1: AWS Glue

NEW QUESTION 10

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 13

A company has used an Amazon Redshift table that is named Orders for 6 months. The company performs weekly updates and deletes on the table. The table has an interleaved sort key on a column that contains AWS Regions. The company wants to reclaim disk space so that the company will not run out of storage space. The company also wants to analyze the sort key column. Which Amazon Redshift command will meet these requirements?

- A. VACUUM FULL Orders
- B. VACUUM DELETE ONLY Orders
- C. VACUUM REINDEX Orders
- D. VACUUM SORT ONLY Orders

Answer: C

Explanation:

Amazon Redshift is a fully managed, petabyte-scale data warehouse service that enables fast and cost-effective analysis of large volumes of data. Amazon Redshift uses columnar storage, compression, and zone maps to optimize the storage and performance of data. However, over time, as data is inserted, updated, or deleted, the physical storage of data can become fragmented, resulting in wasted disk space and degraded query performance. To address this issue, Amazon Redshift provides the VACUUM command, which reclaims disk space and resorts rows in either a specified table or all tables in the current schema¹. The VACUUM command has four options: FULL, DELETE ONLY, SORT ONLY, and REINDEX. The option that best meets the requirements of the question is VACUUM REINDEX, which re-sorts the rows in a table that has an interleaved sort key and rewrites the table to a new location on disk. An interleaved sort key is a type of sort key that gives equal weight to each column in the sort key, and stores the rows in a way that optimizes the performance of queries that filter by multiple columns in the sort key. However, as data is added or changed, the interleaved sort order can become skewed, resulting in suboptimal query performance. The VACUUM REINDEX option restores the optimal interleaved sort order and reclaims disk space by removing deleted rows. This option also analyzes the sort key column and updates the table statistics, which are used by the query optimizer to generate the most efficient query execution plan²³.

The other options are not optimal for the following reasons:

- ? A. VACUUM FULL Orders. This option reclaims disk space by removing deleted rows and resorts the entire table. However, this option is not suitable for tables that have an interleaved sort key, as it does not restore the optimal interleaved sort order. Moreover, this option is the most resource-intensive and time-consuming, as it rewrites the entire table to a new location on disk.
- ? B. VACUUM DELETE ONLY Orders. This option reclaims disk space by removing deleted rows, but does not resort the table. This option is not suitable for tables that have any sort key, as it does not improve the query performance by restoring the sort order. Moreover, this option does not analyze the sort key column and update the table statistics.
- ? D. VACUUM SORT ONLY Orders. This option resorts the entire table, but does not reclaim disk space by removing deleted rows. This option is not suitable for tables that have an interleaved sort key, as it does not restore the optimal interleaved sort order. Moreover, this option does not analyze the sort key column and update the table statistics.

References:

- ? 1: Amazon Redshift VACUUM
- ? 2: Amazon Redshift Interleaved Sorting
- ? 3: Amazon Redshift ANALYZE

NEW QUESTION 14

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 loss¹. AWS DMS supports migration between 20-plus database and analytics engines, such as Microsoft SQL Server to Amazon RDS for SQL Server². 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 monitoring¹. AWS DMS is a cost-effective solution, as you only pay for the compute resources and additional log storage used during the migration process². 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 15

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 needed¹².

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 overhead³.

? 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 20

A data engineering team is using an Amazon Redshift data warehouse for operational reporting. The team wants to prevent performance issues that might result from long-running queries. A data engineer must choose a system table in Amazon Redshift to record anomalies when a query optimizer identifies conditions that might indicate performance issues.

Which table views should the data engineer use to meet this requirement?

- A. STL USAGE CONTROL
- B. STL ALERT EVENT LOG
- C. STL QUERY METRICS
- D. STL PLAN INFO

Answer: B

Explanation:

The STL ALERT EVENT LOG table view records anomalies when the query optimizer identifies conditions that might indicate performance issues. These conditions include skewed data distribution, missing statistics, nested loop joins, and broadcasted data. The STL ALERT EVENT LOG table view can help the data engineer to identify and troubleshoot the root causes of performance issues and optimize the query execution plan. The other table views are not relevant for this requirement. STL USAGE CONTROL records the usage limits and quotas for Amazon Redshift resources. STL QUERY METRICS records the execution time and resource consumption of queries. STL PLAN INFO records the query execution plan and the steps involved in each query. References:

- ? STL ALERT EVENT LOG
- ? System Tables and Views
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

NEW QUESTION 25

A data engineer must build an extract, transform, and load (ETL) pipeline to process and load data from 10 source systems into 10 tables that are in an Amazon Redshift database. All the source systems generate .csv, JSON, or Apache Parquet files every 15 minutes. The source systems all deliver files into one Amazon S3 bucket. The file sizes range from 10 MB to 20 GB. The ETL pipeline must function correctly despite changes to the data schema.

Which data pipeline solutions will meet these requirements? (Choose two.)

- A. Use an Amazon EventBridge rule to run an AWS Glue job every 15 minute
- B. Configure the AWS Glue job to process and load the data into the Amazon Redshift tables.
- C. Use an Amazon EventBridge rule to invoke an AWS Glue workflow job every 15 minute
- D. Configure the AWS Glue workflow to have an on-demand trigger that runs an AWS Glue crawler and then runs an AWS Glue job when the crawler finishes running successful
- E. Configure the AWS Glue job to process and load the data into the Amazon Redshift tables.
- F. Configure an AWS Lambda function to invoke an AWS Glue crawler when a file is loaded into the S3 bucket
- G. Configure an AWS Glue job to process and load the data into the Amazon Redshift table
- H. Create a second Lambda function to run the AWS Glue job
- I. Create an Amazon EventBridge rule to invoke the second Lambda function when the AWS Glue crawler finishes running successfully.
- J. Configure an AWS Lambda function to invoke an AWS Glue workflow when a file is loaded into the S3 bucket
- K. Configure the AWS Glue workflow to have an on-demand trigger that runs an AWS Glue crawler and then runs an AWS Glue job when the crawler finishes running successful
- L. Configure the AWS Glue job to process and load the data into the Amazon Redshift tables.
- M. Configure an AWS Lambda function to invoke an AWS Glue job when a file is loaded into the S3 bucket
- N. Configure the AWS Glue job to read the files from the S3 bucket into an Apache Spark DataFrame
- O. Configure the AWS Glue job to also put smaller partitions of the DataFrame into an Amazon Kinesis Data Firehose delivery stream
- P. Configure the delivery stream to load data into the Amazon Redshift tables.

Answer: AB

Explanation:

Using an Amazon EventBridge rule to run an AWS Glue job or invoke an AWS Glue workflow job every 15 minutes are two possible solutions that will meet the requirements. AWS Glue is a serverless ETL service that can process and load data from various sources to various targets, including Amazon Redshift. AWS Glue can handle different data formats, such as CSV, JSON, and Parquet, and also support schema evolution, meaning it can adapt to changes in the data schema over time. AWS Glue can also leverage Apache Spark to perform distributed processing and transformation of large datasets. AWS Glue integrates with Amazon EventBridge, which is a serverless event bus service that can trigger actions based on rules and schedules. By using an Amazon EventBridge rule, you can invoke an AWS Glue job or workflow every 15 minutes, and configure the job or workflow to run an AWS Glue crawler and then load the data into the Amazon Redshift tables. This way, you can build a cost-effective and scalable ETL pipeline that can handle data from 10 source systems and function correctly despite changes to the data schema.

The other options are not solutions that will meet the requirements. Option C, configuring an AWS Lambda function to invoke an AWS Glue crawler when a file is loaded into the S3 bucket, and creating a second Lambda function to run the AWS Glue job, is not a feasible solution, as it would require a lot of Lambda invocations and coordination. AWS Lambda has some limits on the execution time, memory, and concurrency, which can affect the performance and reliability of the ETL pipeline. Option D, configuring an AWS Lambda function to invoke an AWS Glue workflow when a file is loaded into the S3 bucket, is not a necessary solution, as you can use an Amazon EventBridge rule to invoke the AWS Glue workflow directly, without the need for a Lambda function. Option E, configuring an AWS Lambda function to invoke an AWS Glue job when a file is loaded into the S3 bucket, and configuring the AWS Glue job to put smaller partitions of the DataFrame into an Amazon Kinesis Data Firehose delivery stream, is not a cost-effective solution, as it would incur additional costs for Lambda invocations and data delivery. Moreover, using Amazon Kinesis Data Firehose to load data into Amazon Redshift is not suitable for frequent and small batches of data, as it can cause performance issues and data fragmentation. References:

- ? AWS Glue
- ? Amazon EventBridge
- ? Using AWS Glue to run ETL jobs against non-native JDBC data sources
- ? [AWS Lambda quotas]
- ? [Amazon Kinesis Data Firehose quotas]

NEW QUESTION 30

A company uses an Amazon Redshift cluster that runs on RA3 nodes. The company wants to scale read and write capacity to meet demand. A data engineer needs to identify a solution that will turn on concurrency scaling.

Which solution will meet this requirement?

- A. Turn on concurrency scaling in workload management (WLM) for Redshift Serverless workgroups.
- B. Turn on concurrency scaling at the workload management (WLM) queue level in the Redshift cluster.

- C. Turn on concurrency scaling in the settings during the creation of a new Redshift cluster.
- D. Turn on concurrency scaling for the daily usage quota for the Redshift cluster.

Answer: B

Explanation:

Concurrency scaling is a feature that allows you to support thousands of concurrent users and queries, with consistently fast query performance. When you turn on concurrency scaling, Amazon Redshift automatically adds query processing power in seconds to process queries without any delays. You can manage which queries are sent to the concurrency-scaling cluster by configuring WLM queues. To turn on concurrency scaling for a queue, set the Concurrency Scaling mode value to auto. The other options are either incorrect or irrelevant, as they do not enable concurrency scaling for the existing Redshift cluster on RA3 nodes.

References:

- ? Working with concurrency scaling - Amazon Redshift
- ? Amazon Redshift Concurrency Scaling - Amazon Web Services
- ? Configuring concurrency scaling queues - Amazon Redshift
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide (Chapter 6, page 163)

NEW QUESTION 31

A manufacturing company collects sensor data from its factory floor to monitor and enhance operational efficiency. The company uses Amazon Kinesis Data Streams to publish the data that the sensors collect to a data stream. Then Amazon Kinesis Data Firehose writes the data to an Amazon S3 bucket. The company needs to display a real-time view of operational efficiency on a large screen in the manufacturing facility. Which solution will meet these requirements with the LOWEST latency?

- A. Use Amazon Managed Service for Apache Flink (previously known as Amazon Kinesis Data Analytics) to process the sensor data.
- B. Use a connector for Apache Flink to write data to an Amazon Timestream database.
- C. Use the Timestream database as a source to create a Grafana dashboard.
- D. Configure the S3 bucket to send a notification to an AWS Lambda function when any new object is created. Use the Lambda function to publish the data to Amazon Aurora.
- E. Use the Lambda function to publish the data to Amazon Aurora.
- F. Use Aurora as a source to create an Amazon QuickSight dashboard.
- G. Use Amazon Managed Service for Apache Flink (previously known as Amazon Kinesis Data Analytics) to process the sensor data.
- H. Create a new Data Firehose delivery stream to publish data directly to an Amazon Timestream database.
- I. Use the Timestream database as a source to create an Amazon QuickSight dashboard.
- J. Use AWS Glue bookmarks to read sensor data from the S3 bucket in real time.
- K. Publish the data to an Amazon Timestream database.
- L. Use the Timestream database as a source to create a Grafana dashboard.

Answer: C

Explanation:

This solution will meet the requirements with the lowest latency because it uses Amazon Managed Service for Apache Flink to process the sensor data in real time and write it to Amazon Timestream, a fast, scalable, and serverless time series database. Amazon Timestream is optimized for storing and analyzing time series data, such as sensor data, and can handle trillions of events per day with millisecond latency. By using Amazon Timestream as a source, you can create an Amazon QuickSight dashboard that displays a real-time view of operational efficiency on a large screen in the manufacturing facility. Amazon QuickSight is a fully managed business intelligence service that can connect to various data sources, including Amazon Timestream, and provide interactive visualizations and insights.

The other options are not optimal for the following reasons:

- ? A. Use Amazon Managed Service for Apache Flink (previously known as Amazon Kinesis Data Analytics) to process the sensor data. Use a connector for Apache Flink to write data to an Amazon Timestream database. Use the Timestream database as a source to create a Grafana dashboard. This option is similar to option C, but it uses Grafana instead of Amazon QuickSight to create the dashboard. Grafana is an open source visualization tool that can also connect to Amazon Timestream, but it requires additional steps to set up and configure, such as deploying a Grafana server on Amazon EC2, installing the Amazon Timestream plugin, and creating an IAM role for Grafana to access Timestream. These steps can increase the latency and complexity of the solution.
- ? B. Configure the S3 bucket to send a notification to an AWS Lambda function when any new object is created. Use the Lambda function to publish the data to Amazon Aurora. Use Aurora as a source to create an Amazon QuickSight dashboard. This option is not suitable for displaying a real-time view of operational efficiency, as it introduces unnecessary delays and costs in the data pipeline. First, the sensor data is written to an S3 bucket by Amazon Kinesis Data Firehose, which can have a buffering interval of up to 900 seconds. Then, the S3 bucket sends a notification to a Lambda function, which can incur additional invocation and execution time. Finally, the Lambda function publishes the data to Amazon Aurora, a relational database that is not optimized for time series data and can have higher storage and performance costs than Amazon Timestream.
- ? D. Use AWS Glue bookmarks to read sensor data from the S3 bucket in real time. Publish the data to an Amazon Timestream database. Use the Timestream database as a source to create a Grafana dashboard. This option is also not suitable for displaying a real-time view of operational efficiency, as it uses AWS Glue bookmarks to read sensor data from the S3 bucket. AWS Glue bookmarks are a feature that helps AWS Glue jobs and crawlers keep track of the data that has already been processed, so that they can resume from where they left off. However, AWS Glue jobs and crawlers are not designed for real-time data processing, as they can have a minimum frequency of 5 minutes and a variable start-up time. Moreover, this option also uses Grafana instead of Amazon QuickSight to create the dashboard, which can increase the latency and complexity of the solution.

References:

- ? 1: Amazon Managed Streaming for Apache Flink
- ? 2: Amazon Timestream
- ? 3: Amazon QuickSight
- ? : Analyze data in Amazon Timestream using Grafana
- ? : Amazon Kinesis Data Firehose
- ? : Amazon Aurora
- ? : AWS Glue Bookmarks
- ? : AWS Glue Job and Crawler Scheduling

NEW QUESTION 33

A financial company wants to use Amazon Athena to run on-demand SQL queries on a petabyte-scale dataset to support a business intelligence (BI) application. An AWS Glue job that runs during non-business hours updates the dataset once every day. The BI application has a standard data refresh frequency of 1 hour to comply with company policies.

A data engineer wants to cost optimize the company's use of Amazon Athena without adding any additional infrastructure costs.

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

- A. Configure an Amazon S3 Lifecycle policy to move data to the S3 Glacier Deep Archive storage class after 1 day.
- B. Use the query result reuse feature of Amazon Athena for the SQL queries.

- C. Add an Amazon ElastiCache cluster between the BI application and Athena.
- D. Change the format of the files that are in the dataset to Apache Parquet.

Answer: B

Explanation:

The best solution to cost optimize the company's use of Amazon Athena without adding any additional infrastructure costs is to use the query result reuse feature of Amazon Athena for the SQL queries. This feature allows you to run the same query multiple times without incurring additional charges, as long as the underlying data has not changed and the query results are still in the query result location in Amazon S3. This feature is useful for scenarios where you have a petabyte-scale dataset that is updated infrequently, such as once a day, and you have a BI application that runs the same queries repeatedly, such as every hour. By using the query result reuse feature, you can reduce the amount of data scanned by your queries and save on the cost of running Athena. You can enable or disable this feature at the workgroup level or at the individual query level¹.

Option A is not the best solution, as configuring an Amazon S3 Lifecycle policy to move data to the S3 Glacier Deep Archive storage class after 1 day would not cost optimize the company's use of Amazon Athena, but rather increase the cost and complexity. Amazon S3 Lifecycle policies are rules that you can define to automatically transition objects between different storage classes based on specified criteria, such as the age of the object². S3 Glacier Deep Archive is the lowest-cost storage class in Amazon S3, designed

for long-term data archiving that is accessed once or twice in a year³. While moving data to S3 Glacier Deep Archive can reduce the storage cost, it would also increase the retrieval cost and latency, as it takes up to 12 hours to restore the data from S3 Glacier Deep Archive³. Moreover, Athena does not support querying data that is in S3 Glacier or S3 Glacier Deep Archive storage classes⁴. Therefore, using this option would not meet the requirements of running on-demand SQL queries on the dataset.

Option C is not the best solution, as adding an Amazon ElastiCache cluster between the BI application and Athena would not cost optimize the company's use of Amazon Athena, but rather increase the cost and complexity. Amazon ElastiCache is a service that offers fully managed in-memory data stores, such as Redis and Memcached, that can improve the performance and scalability of web applications by caching frequently accessed data. While using ElastiCache can reduce the latency and load on the BI application, it would not reduce the amount of data scanned by Athena, which is the main factor that determines the cost of running Athena. Moreover, using ElastiCache would introduce additional infrastructure costs and operational overhead, as you would have to provision, manage, and scale the ElastiCache cluster, and integrate it with the BI application and Athena. Option D is not the best solution, as changing the format of the files that are in the dataset to Apache Parquet would not cost optimize the company's use of Amazon Athena without adding any additional infrastructure costs, but rather increase the complexity. Apache Parquet is a columnar storage format that can improve the performance of analytical queries by reducing the amount of data that needs to be scanned and providing efficient compression and encoding schemes. However, changing the format of the files that are in the dataset to Apache Parquet would require additional processing and transformation steps, such as using AWS Glue or Amazon EMR to convert the files from their original format to Parquet, and storing the converted files in a separate location in Amazon S3. This would increase the complexity and the operational overhead of the data pipeline, and also incur additional costs for using AWS Glue or Amazon EMR. References:

- ? Query result reuse
- ? Amazon S3 Lifecycle
- ? S3 Glacier Deep Archive
- ? Storage classes supported by Athena
- ? [What is Amazon ElastiCache?]
- ? [Amazon Athena pricing]
- ? [Columnar Storage Formats]
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

NEW QUESTION 34

A data engineer uses Amazon Redshift to run resource-intensive analytics processes once every month. Every month, the data engineer creates a new Redshift provisioned cluster. The data engineer deletes the Redshift provisioned cluster after the analytics processes are complete every month. Before the data engineer deletes the cluster each month, the data engineer unloads backup data from the cluster to an Amazon S3 bucket.

The data engineer needs a solution to run the monthly analytics processes that does not require the data engineer to manage the infrastructure manually. Which solution will meet these requirements with the LEAST operational overhead?

- A. Use Amazon Step Functions to pause the Redshift cluster when the analytics processes are complete and to resume the cluster to run new processes every month.
- B. Use Amazon Redshift Serverless to automatically process the analytics workload.
- C. Use the AWS CLI to automatically process the analytics workload.
- D. Use AWS CloudFormation templates to automatically process the analytics workload.

Answer: B

Explanation:

Amazon Redshift Serverless is a new feature of Amazon Redshift that enables you to run SQL queries on data in Amazon S3 without provisioning or managing any clusters. You can use Amazon Redshift Serverless to automatically process the analytics workload, as it scales up and down the compute resources based on the query demand, and charges you only for the resources consumed. This solution will meet the requirements with the least operational overhead, as it does not require the data engineer to create, delete, pause, or resume any Redshift clusters, or to manage any infrastructure manually. You can use the Amazon Redshift Data API to run queries from the AWS CLI, AWS SDK, or AWS Lambda functions¹².

The other options are not optimal for the following reasons:

- ? A. Use Amazon Step Functions to pause the Redshift cluster when the analytics processes are complete and to resume the cluster to run new processes every month. This option is not recommended, as it would still require the data engineer to create and delete a new Redshift provisioned cluster every month, which can incur additional costs and time. Moreover, this option would require the data engineer to use Amazon Step Functions to orchestrate the workflow of pausing and resuming the cluster, which can add complexity and overhead.
- ? C. Use the AWS CLI to automatically process the analytics workload. This option is vague and does not specify how the AWS CLI is used to process the analytics workload. The AWS CLI can be used to run queries on data in Amazon S3 using Amazon Redshift Serverless, Amazon Athena, or Amazon EMR, but each of these services has different features and benefits. Moreover, this option does not address the requirement of not managing the infrastructure manually, as the data engineer may still need to provision and configure some resources, such as Amazon EMR clusters or Amazon Athena workgroups.
- ? D. Use AWS CloudFormation templates to automatically process the analytics workload. This option is also vague and does not specify how AWS CloudFormation templates are used to process the analytics workload. AWS CloudFormation is a service that lets you model and provision AWS resources using templates. You can use AWS CloudFormation templates to create and delete a Redshift provisioned cluster every month, or to create and configure other AWS resources, such as Amazon EMR, Amazon Athena, or Amazon Redshift Serverless. However, this option does not address the requirement of not managing the infrastructure manually, as the data engineer may still need to write and maintain the AWS CloudFormation templates, and to monitor the status and performance of the resources.

References:

- ? 1: Amazon Redshift Serverless
- ? 2: Amazon Redshift Data API
- ? : Amazon Step Functions
- ? : AWS CLI

? : AWS CloudFormation

NEW QUESTION 37

A company has multiple applications that use datasets that are stored in an Amazon S3 bucket. The company has an ecommerce application that generates a dataset that contains personally identifiable information (PII). The company has an internal analytics application that does not require access to the PII. To comply with regulations, the company must not share PII unnecessarily. A data engineer needs to implement a solution that with redact PII dynamically, based on the needs of each application that accesses the dataset.

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

- A. Create an S3 bucket policy to limit the access each application has
- B. Create multiple copies of the dataset
- C. Give each dataset copy the appropriate level of redaction for the needs of the application that accesses the copy.
- D. Create an S3 Object Lambda endpoint
- E. Use the S3 Object Lambda endpoint to read data from the S3 bucket
- F. Implement redaction logic within an S3 Object Lambda function to dynamically redact PII based on the needs of each application that accesses the data.
- G. Use AWS Glue to transform the data for each application
- H. Create multiple copies of the dataset
- I. Give each dataset copy the appropriate level of redaction for the needs of the application that accesses the copy.
- J. Create an API Gateway endpoint that has custom authorizer
- K. Use the API Gateway endpoint to read data from the S3 bucket
- L. Initiate a REST API call to dynamically redact PII based on the needs of each application that accesses the data.

Answer: B

Explanation:

Option B is the best solution to meet the requirements with the least operational overhead because S3 Object Lambda is a feature that allows you to add your own code to process data retrieved from S3 before returning it to an application. S3 Object Lambda works with S3 GET requests and can modify both the object metadata and the object data. By using S3 Object Lambda, you can implement redaction logic within an S3 Object Lambda function to dynamically redact PII based on the needs of each application that accesses the data. This way, you can avoid creating and maintaining multiple copies of the dataset with different levels of redaction.

Option A is not a good solution because it involves creating and managing multiple copies of the dataset with different levels of redaction for each application. This option adds complexity and storage cost to the data protection process and requires additional resources and configuration. Moreover, S3 bucket policies cannot enforce fine-grained data access control at the row and column level, so they are not sufficient to redact PII.

Option C is not a good solution because it involves using AWS Glue to transform the data for each application. AWS Glue is a fully managed service that can extract, transform, and load (ETL) data from various sources to various destinations, including S3. AWS Glue can also convert data to different formats, such as Parquet, which is a columnar storage format that is optimized for analytics. However, in this scenario, using AWS Glue to redact PII is not the best option because it requires creating and maintaining multiple copies of the dataset with different levels of redaction for each application. This option also adds extra time and cost to the data protection process and requires additional resources and configuration.

Option D is not a good solution because it involves creating and configuring an API Gateway endpoint that has custom authorizers. API Gateway is a service that allows you to create, publish, maintain, monitor, and secure APIs at any scale. API Gateway can also integrate with other AWS services, such as Lambda, to provide custom logic for processing requests. However, in this scenario, using API Gateway to redact PII is not the best option because it requires writing and maintaining custom code and configuration for the API endpoint, the custom authorizers, and the REST API call. This option also adds complexity and latency to the data protection process and requires additional resources and configuration.

References:

- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide
- ? Introducing Amazon S3 Object Lambda – Use Your Code to Process Data as It Is Being Retrieved from S3
- ? Using Bucket Policies and User Policies - Amazon Simple Storage Service
- ? AWS Glue Documentation
- ? What is Amazon API Gateway? - Amazon API Gateway

NEW QUESTION 40

A retail company has a customer data hub in an Amazon S3 bucket. Employees from many countries use the data hub to support company-wide analytics. A governance team must ensure that the company's data analysts can access data only for customers who are within the same country as the analysts.

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

- A. Create a separate table for each country's customer data
- B. Provide access to each analyst based on the country that the analyst serves.
- C. Register the S3 bucket as a data lake location in AWS Lake Formation
- D. Use the Lake Formation row-level security features to enforce the company's access policies.
- E. Move the data to AWS Regions that are close to the countries where the customers are
- F. Provide access to each analyst based on the country that the analyst serves.
- G. Load the data into Amazon Redshift
- H. Create a view for each country
- I. Create separate IAM roles for each country to provide access to data from each country
- J. Assign the appropriate roles to the analysts.

Answer: B

Explanation:

AWS Lake Formation is a service that allows you to easily set up, secure, and manage data lakes. One of the features of Lake Formation is row-level security, which enables you to control access to specific rows or columns of data based on the identity or role of the user. This feature is useful for scenarios where you need to restrict access to sensitive or regulated data, such as customer data from different countries. By registering the S3 bucket as a data lake location in Lake Formation, you can use the Lake Formation console or APIs to define and apply row-level security policies to the data in the bucket. You can also use Lake Formation blueprints to automate the ingestion and transformation of data from various sources into the data lake. This solution requires the least operational effort compared to the other options, as it does not involve creating or moving data, or managing multiple tables, views, or roles. References:

- ? AWS Lake Formation
- ? Row-Level Security
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 4: Data Lakes and Data Warehouses, Section 4.2: AWS Lake Formation

NEW QUESTION 41

A data engineer is using Amazon Athena to analyze sales data that is in Amazon S3. The data engineer writes a query to retrieve sales amounts for 2023 for several products from a table named sales_data. However, the query does not return results for all of the products that are in the sales_data table. The data engineer needs to troubleshoot the query to resolve the issue.

The data engineer's original query is as follows: SELECT product_name, sum(sales_amount) FROM sales_data WHERE year = 2023 GROUP BY product_name

How should the data engineer modify the Athena query to meet these requirements?

- A. Replace sum(sales amount) with count(*) for the aggregation.
- B. Change WHERE year = 2023 to WHERE extract(year FROM sales_data) = 2023.
- C. Add HAVING sum(sales amount) > 0 after the GROUP BY clause.
- D. Remove the GROUP BY clause

Answer: B

Explanation:

The original query does not return results for all of the products because the year column in the sales_data table is not an integer, but a timestamp. Therefore, the WHERE clause does not filter the data correctly, and only returns the products that have a null value for the year column. To fix this, the data engineer should use the extract function to extract the year from the timestamp and compare it with 2023. This way, the query will return the correct results for all of the products in the sales_data table. The other options are either incorrect or irrelevant, as they do not address the root cause of the issue. Replacing sum with count does not change the filtering condition, adding HAVING clause does not affect the grouping logic, and removing the GROUP BY clause does not solve the problem of missing products. References:

? Troubleshooting JSON queries - Amazon Athena (Section: JSON related errors)

? When I query a table in Amazon Athena, the TIMESTAMP result is empty (Section: Resolution)

? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide (Chapter 7, page 197)

NEW QUESTION 42

A data engineer has a one-time task to read data from objects that are in Apache Parquet format in an Amazon S3 bucket. The data engineer needs to query only one column of the data.

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

- A. Configure an AWS Lambda function to load data from the S3 bucket into a pandas dataframe- Write a SQL SELECT statement on the dataframe to query the required column.
- B. Use S3 Select to write a SQL SELECT statement to retrieve the required column from the S3 objects.
- C. Prepare an AWS Glue DataBrew project to consume the S3 objects and to query the required column.
- D. Run an AWS Glue crawler on the S3 object
- E. Use a SQL SELECT statement in Amazon Athena to query the required column.

Answer: B

Explanation:

Option B is the best solution to meet the requirements with the least operational overhead because S3 Select is a feature that allows you to retrieve only a subset of data from an S3 object by using simple SQL expressions. S3 Select works on objects stored in CSV, JSON, or Parquet format. By using S3 Select, you can avoid the need to download and process the entire S3 object, which reduces the amount of data transferred and the computation time. S3 Select is also easy to use and does not require any additional services or resources.

Option A is not a good solution because it involves writing custom code and configuring an AWS Lambda function to load data from the S3 bucket into a pandas dataframe and query the required column. This option adds complexity and latency to the data retrieval process and requires additional resources and configuration. Moreover, AWS Lambda has limitations on the execution time, memory, and concurrency, which may affect the performance and reliability of the data retrieval process.

Option C is not a good solution because it involves creating and running an AWS Glue DataBrew project to consume the S3 objects and query the required column. AWS Glue DataBrew is a visual data preparation tool that allows you to clean, normalize, and transform data without writing code. However, in this scenario, the data is already in Parquet format, which is a columnar storage format that is optimized for analytics. Therefore, there is no need to use AWS Glue DataBrew to prepare the data. Moreover, AWS Glue DataBrew adds extra time and cost to the data retrieval process and requires additional resources and configuration.

Option D is not a good solution because it involves running an AWS Glue crawler on the S3 objects and using a SQL SELECT statement in Amazon Athena to query the required column. 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. Amazon Athena is a serverless interactive query service that allows you to analyze data in S3 using standard SQL. However, in this scenario, the schema and format of the data are already known and fixed, so there is no need to run a crawler to discover them. Moreover, running a crawler and using Amazon Athena adds extra time and cost to the data retrieval process and requires additional services and configuration.

References:

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

? S3 Select and Glacier Select - Amazon Simple Storage Service

? AWS Lambda - FAQs

? What Is AWS Glue DataBrew? - AWS Glue DataBrew

? Populating the AWS Glue Data Catalog - AWS Glue

? What is Amazon Athena? - Amazon Athena

NEW QUESTION 44

A data engineer needs to maintain a central metadata repository that users access through Amazon EMR and Amazon Athena queries. The repository needs to provide the schema and properties of many tables. Some of the metadata is stored in Apache Hive. The data engineer needs to import the metadata from Hive into the central metadata repository.

Which solution will meet these requirements with the LEAST development effort?

- A. Use Amazon EMR and Apache Ranger.
- B. Use a Hive metastore on an EMR cluster.
- C. Use the AWS Glue Data Catalog.
- D. Use a metastore on an Amazon RDS for MySQL DB instance.

Answer: C

Explanation:

The AWS Glue Data Catalog is an Apache Hive metastore-compatible catalog that provides a central metadata repository for various data sources and formats. You can use the AWS Glue Data Catalog as an external Hive metastore for Amazon EMR and Amazon Athena queries, and import metadata from existing Hive metastores into the Data Catalog. This solution requires the least development effort, as you can use AWS Glue crawlers to automatically discover and catalog the metadata from Hive, and use the AWS Glue console, AWS CLI, or Amazon EMR API to configure the Data Catalog as the Hive metastore. The other options are either more complex or require additional steps, such as setting up Apache Ranger for security, managing a Hive metastore on an EMR cluster or an RDS instance, or migrating the metadata manually. References:

- ? Using the AWS Glue Data Catalog as the metastore for Hive (Section: Specifying AWS Glue Data Catalog as the metastore)
- ? Metadata Management: Hive Metastore vs AWS Glue (Section: AWS Glue Data Catalog)
- ? AWS Glue Data Catalog support for Spark SQL jobs (Section: Importing metadata from an existing Hive metastore)
- ? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide (Chapter 5, page 131)

NEW QUESTION 47

.....

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](#)