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 1AM policy that uses the tags to apply appropriate permissions to the workgroup.
G. Create an JAM 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 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 Bl cluster asa 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 wee
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 anAmazon 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 classe
B. Create S3 Lifecycle policies for the S3 buckets to move objects to cost-optimized storage classe
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 infrequentl
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-Tierin
G. Activate the Deep Archive Access tier.
H. Use S3 Intelligent-Tierin
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 needs to schedule a workflow that runs a set of AWS Glue jobs every day. The data engineer does not require the Glue jobs to run or finish at a specific time.
Which solution will run the Glue jobs in the MOST cost-effective way?

A. Choose the FLEX execution class in the Glue job properties.
B. Use the Spot Instance type in Glue job properties.
C. Choose the STANDARD execution class in the Glue job properties.
D. Choose the latest version in the GlueVersion field in the Glue job properties.

Answer: A

Explanation:
The FLEX execution class allows you to run AWS Glue jobs on spare compute capacity instead of dedicated hardware. This can reduce the cost of running non-urgent or non-time sensitive data integration workloads, such as testing and one-time data loads. The FLEX execution class is available for AWS Glue 3.0 Spark jobs. The other options are not as cost-effective as FLEX, because they either use dedicated resources (STANDARD) or do not affect the cost at all (Spot Instance type and GlueVersion). References:
? Introducing AWS Glue Flex jobs: Cost savings on ETL workloads
? Serverless Data Integration – AWS Glue Pricing
? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide (Chapter 5, page 125)

NEW QUESTION 5
A company created an extract, transform, and load (ETL) data pipeline in AWS Glue. A data engineer must crawl a table that is in Microsoft SQL Server. The data engineer needs to extract, transform, and load the output of the crawl to an Amazon S3 bucket. The data engineer also must orchestrate the data pipeline.
Which AWS service or feature will meet these requirements MOST cost-effectively?

A. AWS Step Functions
B. AWS Glue workflows
C. AWS Glue Studio
D. Amazon Managed Workflows for Apache Airflow (Amazon MWAA)

Answer: B

Explanation:
AWS Glue workflows are a cost-effective way to orchestrate complex ETL jobs that involve multiple crawlers, jobs, and triggers. AWS Glue workflows allow you to visually monitor the progress and dependencies of your ETL tasks, and automatically handle errors and retries. AWS Glue workflows also integrate with other AWS services, such as Amazon S3, Amazon Redshift, and AWS Lambda, among others, enabling you to leverage these services for your data processing workflows. AWS Glue workflows are serverless, meaning you only pay for the resources you use, and you don’t have to manage any infrastructure.
AWS Step Functions, AWS Glue Studio, and Amazon MWAA are also possible options for orchestrating ETL pipelines, but they have some drawbacks compared to AWS Glue workflows. AWS Step Functions is a serverless function orchestrator that can handle different types of data processing, such as real-time, batch, and stream processing. However, AWS Step Functions requires you to write code to define your state machines, which can be complex and error-prone. AWS Step Functions also charges you for every state transition, which can add up quickly for large-scale ETL pipelines.
AWS Glue Studio is a graphical interface that allows you to create and run AWS Glue ETL jobs without writing code. AWS Glue Studio simplifies the process of building, debugging, and monitoring your ETL jobs, and provides a range of pre-built transformations and connectors. However, AWS Glue Studio does not support workflows, meaning you cannot orchestrate multiple ETL jobs or crawlers with dependencies and triggers. AWS Glue Studio also does not support streaming data sources or targets, which limits its use cases for real-time data processing.
Amazon MWAA is a fully managed service that makes it easy to run open-source versions of Apache Airflow on AWS and build workflows to run your ETL jobs and data pipelines. Amazon MWAA provides a familiar and flexible environment for data engineers who are familiar with Apache Airflow, and integrates with a range of AWS services such as Amazon EMR, AWS Glue, and AWS Step Functions. However, Amazon MWAA is not serverless, meaning you have to provision and pay for the resources you need, regardless of your usage. Amazon MWAA also requires you to write code to define your DAGs, which can be challenging and time-consuming for complex ETL pipelines. References:
? AWS Glue Workflows
? AWS Step Functions
? AWS Glue Studio
? Amazon MWAA
? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

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 data engineer is building a data pipeline on AWS by using AWS Glue extract, transform, and load (ETL) jobs. The data engineer needs to process data from Amazon RDS and MongoDB, perform transformations, and load the transformed data into Amazon Redshift for analytics. The data updates must occur every hour.
Which combination of tasks will meet these requirements with the LEAST operational overhead? (Choose two.)

A. Configure AWS Glue triggers to run the ETL jobs even/ hour.
B. Use AWS Glue DataBrewto clean and prepare the data for analytics.
C. Use AWS Lambda functions to schedule and run the ETL jobs even/ hour.
D. Use AWS Glue connections to establish connectivity between the data sources and Amazon Redshift.
E. Use the Redshift Data API to load transformed data into Amazon Redshift.

Answer: AD

Explanation:
The correct answer is to configure AWS Glue triggers to run the ETL jobs every hour and use AWS Glue connections to establish connectivity between the data sources and Amazon Redshift. AWS Glue triggers are a way to schedule and orchestrate ETL jobs with the least operational overhead. AWS Glue connections are a way to securely connect to data sources and targets using JDBC or MongoDB drivers. AWS Glue DataBrew is a visual data preparation tool that does not support MongoDB as a data source. AWS Lambda functions are a serverless option to schedule and run ETL jobs, but they have a limit of 15 minutes for execution time, which may not be enough for complex transformations. The Redshift Data API is a way to run SQL commands on Amazon Redshift clusters without needing a persistent connection, but it does not support loading data from AWS Glue ETL jobs. References:
? AWS Glue triggers
? AWS Glue connections
? AWS Glue DataBrew
? [AWS Lambda functions]
? [Redshift Data API]

NEW QUESTION 8
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 to the permissions-related errors? (Choose two.)

A. There is no connection between QuickSgqht and Athena.
B. The Athena tables are not cataloged.
C. QuickSiqht does not have access to the S3 bucket.
D. QuickSight does not have access to decrypt S3 data.
E. There is no 1AM role assigned to QuickSiqht.

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 9
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 10
A company uses Amazon RDS for MySQL as the database for a critical application. The database workload is mostly writes, with a small number of reads.
A data engineer notices that the CPU utilization of the DB instance is very high. The high CPU utilization is slowing down the application. The data engineer must reduce the CPU utilization of the DB Instance.
Which actions should the data engineer take to meet this requirement? (Choose two.)

A. Use the Performance Insights feature of Amazon RDS to identify queries that have high CPU utilizatio
B. Optimize the problematic queries.
C. Modify the database schema to include additional tables and indexes.
D. Reboot the RDS DB instance once each week.
E. Upgrade to a larger instance size.
F. Implement caching to reduce the database query load.

Answer: AE

Explanation:
Amazon RDS is a fully managed service that provides relational databases in the cloud. Amazon RDS for MySQL is one of the supported database engines that you can use to run your applications. Amazon RDS provides various features and tools to monitor and optimize the performance of your DB instances, such as Performance Insights, Enhanced Monitoring, CloudWatch metrics and alarms, etc.
Using the Performance Insights feature of Amazon RDS to identify queries that have high CPU utilization and optimizing the problematic queries will help reduce the CPU utilization of the DB instance. Performance Insights is a feature that allows you to analyze the load on your DB instance and determine what is causing performance issues. Performance Insights collects, analyzes, and displays database performance data using an interactive dashboard. You can use Performance Insights to identify the top SQL statements, hosts, users, or processes that are consuming the most CPU resources. You can also drill down into the details of each query and see the execution plan, wait events, locks, etc. By using Performance Insights, you can pinpoint the root cause of the high CPU utilization and optimize the queries accordingly. For example, you can rewrite the queries to make them more efficient, add or remove indexes, use prepared statements, etc.
Implementing caching to reduce the database query load will also help reduce the CPU utilization of the DB instance. Caching is a technique that allows you to store frequently accessed data in a fast and scalable storage layer, such as Amazon ElastiCache. By using caching, you can reduce the number of requests that hit your database, which in turn reduces the CPU load on your DB instance. Caching also improves the performance and availability of your application, as it reduces the latency and increases the throughput of your data access. You can use caching for various scenarios, such as storing session data, user preferences, application configuration, etc. You can also use caching for read-heavy workloads, such as displaying product details, recommendations, reviews, etc.
The other options are not as effective as using Performance Insights and caching. Modifying the database schema to include additional tables and indexes may or may not improve the CPU utilization, depending on the nature of the workload and the queries. Adding more tables and indexes may increase the complexity and overhead of the database, which may negatively affect the performance. Rebooting the RDS DB instance once each week will not reduce the CPU utilization, as it will not address the underlying cause of the high CPU load. Rebooting may also cause downtime and disruption to your application. Upgrading to a larger instance size may reduce the CPUutilization, but it will also increase the cost and complexity of your solution. Upgrading may also not be necessary if you can optimize the queries and reduce the database load by using caching. References:
? Amazon RDS
? Performance Insights
? Amazon ElastiCache
? [AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide], Chapter 3: Data Storage and Management, Section 3.1: Amazon RDS

NEW QUESTION 10
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 functio
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 cal
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 successfull

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 13
A company uses Amazon Redshift for its data warehouse. The company must automate refresh schedules for Amazon Redshift materialized views.
Which solution will meet this requirement with the LEAST effort?

A. Use Apache Airflow to refresh the materialized views.
B. Use an AWS Lambda user-defined function (UDF) within Amazon Redshift to refresh the materialized views.
C. Use the query editor v2 in Amazon Redshift to refresh the materialized views.
D. Use an AWS Glue workflow to refresh the materialized views.

Answer: C

Explanation:
The query editor v2 in Amazon Redshift is a web-based tool that allows users to run SQL queries and scripts on Amazon Redshift clusters. The query editor v2 supports creating and managing materialized views, which are precomputed results of a query that can improve the performance of subsequent queries. The query editor v2 also supports scheduling queries to run at specified intervals, which can be used to refresh materialized views automatically. This solution requires the least effort, as it does not involve any additional services, coding, or configuration. The other solutions are more complex and require more operational overhead. Apache Airflow is an open-source platform for orchestrating workflows, which can be used to refresh materialized views, but it requires setting up and managing an Airflow environment, creating DAGs (directed acyclic graphs) to define the workflows, and integrating with Amazon Redshift. AWS Lambda is a serverless compute service that can run code in response to events, which can be used to refresh materialized views, but it requires creating and deploying Lambda functions, defining UDFs within Amazon Redshift, and triggering the functions using events or schedules. AWS Glue is a fully managed ETL service that can run jobs to transform and load data, which can be used to refresh materialized views, but it requires creating and configuring Glue jobs, defining Glue workflows to orchestrate the jobs, and scheduling the workflows using triggers. References:
? Query editor V2
? Working with materialized views
? Scheduling queries
? [AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide]

NEW QUESTION 15
A company has a frontend ReactJS website that uses Amazon API Gateway to invoke REST APIs. The APIs perform the functionality of the website. A data engineer needs to write a Python script that can be occasionally invoked through API Gateway. The code must return results to API Gateway.
Which solution will meet these requirements with the LEAST operational overhead?

A. Deploy a custom Python script on an Amazon Elastic Container Service (Amazon ECS) cluster.
B. Create an AWS Lambda Python function with provisioned concurrency.
C. Deploy a custom Python script that can integrate with API Gateway on Amazon Elastic Kubernetes Service (Amazon EKS).
D. Create an AWS Lambda functio
E. Ensure that the function is warm byscheduling an Amazon EventBridge rule to invoke the Lambda function every 5 minutes by usingmock events.

Answer: B

Explanation:
AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers. You can use Lambda to create functions that perform custom logic and integrate with other AWS services, such as API Gateway. Lambda automatically scales your application by running code in response to each trigger. You pay only for the compute time you consume1.
Amazon ECS is a fully managed container orchestration service that allows you to run and scale containerized applications on AWS. You can use ECS to deploy, manage, and scale Docker containers using either Amazon EC2 instances or AWS Fargate, a serverless compute engine for containers2.
Amazon EKS is a fully managed Kubernetes service that allows you to run Kubernetes clusters on AWS without needing to install, operate, or maintain your own Kubernetes control plane. You can use EKS to deploy, manage, and scale containerized applications using Kubernetes on AWS3.
The solution that meets the requirements with the least operational overhead is to create an AWS Lambda Python function with provisioned concurrency. This solution has the following advantages:
? It does not require you to provision, manage, or scale any servers or clusters, as Lambda handles all the infrastructure for you. This reduces the operational complexity and cost of running your code.
? It allows you to write your Python script as a Lambda function and integrate it with API Gateway using a simple configuration. API Gateway can invoke your Lambda function synchronously or asynchronously, and return the results to the frontend website.
? It ensures that your Lambda function is ready to respond to API requests without any cold start delays, by using provisioned concurrency. Provisioned concurrency is a feature that keeps your function initialized and hyper-ready to respond in double-digit milliseconds. You can specify the number of concurrent executions that you want to provision for your function.

Option A is incorrect because it requires you to deploy a custom Python script on an Amazon ECS cluster. This solution has the following disadvantages:
? It requires you to provision, manage, and scale your own ECS cluster, either using EC2 instances or Fargate. This increases the operational complexity and cost of running your code.
? It requires you to package your Python script as a Docker container image and store it in a container registry, such as Amazon ECR or Docker Hub. This adds an extra step to your deployment process.
? It requires you to configure your ECS cluster to integrate with API Gateway, either using an Application Load Balancer or a Network Load Balancer. This adds another layer of complexity to your architecture.
Option C is incorrect because it requires you to deploy a custom Python script that can integrate with API Gateway on Amazon EKS. This solution has the following disadvantages:
? It requires you to provision, manage, and scale your own EKS cluster, either using EC2 instances or Fargate. This increases the operational complexity and cost of running your code.
? It requires you to package your Python script as a Docker container image and store it in a container registry, such as Amazon ECR or Docker Hub. This adds an extra step to your deployment process.
? It requires you to configure your EKS cluster to integrate with API Gateway, either using an Application Load Balancer, a Network Load Balancer, or a service of type LoadBalancer. This adds another layer of complexity to your architecture.
Option D is incorrect because it requires you to create an AWS Lambda function and ensure that the function is warm by scheduling an Amazon EventBridge rule to invoke the Lambda function every 5 minutes by using mock events. This solution has the following disadvantages:
? It does not guarantee that your Lambda function will always be warm, as Lambda may scale down your function if it does not receive any requests for a long period of time. This may cause cold start delays when your function is invoked by API Gateway.
? It incurs unnecessary costs, as you pay for the compute time of your Lambda function every time it is invoked by the EventBridge rule, even if it does not perform any useful work1.
References:
? 1: AWS Lambda - Features
? 2: Amazon Elastic Container Service - Features
? 3: Amazon Elastic Kubernetes Service - Features
? [4]: Building API Gateway REST API with Lambda integration - Amazon API Gateway
? [5]: Improving latency with Provisioned Concurrency - AWS Lambda
? [6]: Integrating Amazon ECS with Amazon API Gateway - Amazon Elastic Container Service
? [7]: Integrating Amazon EKS with Amazon API Gateway - Amazon Elastic Kubernetes Service
? [8]: Managing concurrency for a Lambda function - AWS Lambda

NEW QUESTION 20
A company stores data from an application in an Amazon DynamoDB table that operates in provisioned capacity mode. The workloads of the application have predictable throughput load on a regular schedule. Every Monday, there is an immediate increase in activity early in the morning. The application has very low usage during weekends.
The company must ensure that the application performs consistently during peak usage times
Which solution will meet these requirements in the MOST cost-effective way?

A. Increase the provisioned capacity to the maximum capacity that is currently present during peak load times.
B. Divide the table into two table
C. Provision each table with half of the provisioned capacity of the original tabl
D. Spread queries evenly across both tables.
E. Use AWS Application Auto Scaling to schedule higher provisioned capacity for peak usage time
F. Schedule lower capacity during off-peak times.
G. Change the capacity mode from provisioned to on-deman
H. Configure the table to scale up and scale down based on the load on the table.

Answer: C

Explanation:
Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. DynamoDB offers two capacity modes for throughput capacity: provisioned and on-demand. In provisioned capacity mode, you specify the number of read and write capacity units per second that you expect your application to require. DynamoDB reserves the resources to meet your throughput needs with consistent performance. In on-demand capacity mode, you pay per request and DynamoDB scales the resources up and down automatically based on the actual workload. On-demand capacity mode is suitable for unpredictable workloads that can vary significantly over time1.
The solution that meets the requirements in the most cost-effective way is to use AWS Application Auto Scaling to schedule higher provisioned capacity for peak usage times and lower capacity during off-peak times. This solution has the following advantages:
? It allows you to optimize the cost and performance of your DynamoDB table by adjusting the provisioned capacity according to your predictable workload patterns. You can use scheduled scaling to specify the date and time for the scaling actions, and the new minimum and maximum capacity limits. For example, you can schedule higher capacity for every Monday morning and lower capacity for weekends2.
? It enables you to take advantage of the lower cost per unit of provisioned capacity mode compared to on-demand capacity mode. Provisioned capacity mode charges a flat hourly rate for the capacity you reserve, regardless of how much you use. On-demand capacity mode charges for each read and write request you consume, with nominimum capacity required. For predictable workloads, provisioned capacity mode can be more cost-effective than on-demand capacity mode1.
? It ensures that your application performs consistently during peak usage times by having enough capacity to handle the increased load. You can also use auto scaling to automatically adjust the provisioned capacity based on the actual utilization of your table, and set a target utilization percentage for your table or global secondary index. This way, you can avoid under-provisioning or over- provisioning your table2.
Option A is incorrect because it suggests increasing the provisioned capacity to the maximum capacity that is currently present during peak load times. This solution has the following disadvantages:
? It wastes money by paying for unused capacity during off-peak times. If you provision the same high capacity for all times, regardless of the actual workload, you are over-provisioning your table and paying for resources that you don’t need1.
? It does not account for possible changes in the workload patterns over time. If your peak load times increase or decrease in the future, you may need to manually adjust the provisioned capacity to match the new demand. This adds operational overhead and complexity to your application2.
Option B is incorrect because it suggests dividing the table into two tables and provisioning each table with half of the provisioned capacity of the original table. This solution has the following disadvantages:
? It complicates the data model and the application logic by splitting the data into two
separate tables. You need to ensure that the queries are evenly distributed across both tables, and that the data is consistent and synchronized between them. This adds extra development and maintenance effort to your application3.
? It does not solve the problem of adjusting the provisioned capacity according to the workload patterns. You still need to manually or automatically scale the capacity of each table based on the actual utilization and demand. This may result in under- provisioning or over-provisioning your tables2.
Option D is incorrect because it suggests changing the capacity mode from provisioned to on-demand. This solution has the following disadvantages:
? It may incur higher costs than provisioned capacity mode for predictable workloads. On-demand capacity mode charges for each read and write request you consume, with no minimum capacity required. For predictable workloads, provisioned capacity mode can be more cost-effective than on-demand capacity mode,

as you can reserve the capacity you need at a lower rate1.
? It may not provide consistent performance during peak usage times, as on-demand capacity mode may take some time to scale up the resources to meet the sudden increase in demand. On-demand capacity mode uses adaptive capacity to handle bursts of traffic, but it may not be able to handle very large spikes or sustained high throughput. In such cases, you may experience throttling or increased latency.
References:
? 1: Choosing the right DynamoDB capacity mode - Amazon DynamoDB
? 2: Managing throughput capacity automatically with DynamoDB auto scaling - Amazon DynamoDB
? 3: Best practices for designing and using partition keys effectively - Amazon DynamoDB
? [4]: On-demand mode guidelines - Amazon DynamoDB
? [5]: How to optimize Amazon DynamoDB costs - AWS Database Blog
? [6]: DynamoDB adaptive capacity: How it works and how it helps - AWS Database Blog
? [7]: Amazon DynamoDB pricing - Amazon Web Services (AWS)

NEW QUESTION 25
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 bucke
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 bucke
D. Create a pipeline in Apache Spark.
E. Create a PvSpark proqram 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 tabl
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 28
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 noteboo
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 fil
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 assets1.
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 DataBrewto 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 column2.
? 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 recipes1.
? 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 sources3.
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 33
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 downtown 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 overmany 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 35
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, suchas 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 37
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 38
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 accoun
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 accoun
D. Create an IAM role and a trust policy to grant CloudWatch Logs the permission to put data into the strea
E. Create a subscription filter in the security AWS account.
F. Create a destination data stream in the production AWS accoun
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 accoun
I. Create an IAM role and a trust policy to grant CloudWatch Logs the permission to put data into the strea
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 40
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 .csvdata to JSON format by fetchingonly 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 42
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 dat
B. Use a connector for Apache Flink to write data to an Amazon Timestream databas
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 create
E. Use the Lambda function to publish the data to Amazon Auror
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 dat
H. Create a new Data Firehose delivery stream to publish data directly to an Amazon Timestream databas
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 tim
K. Publish the data to an Amazon Timestream databas
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 AmazonTimestream 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 insights123.
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 44
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 45
A company currently stores all of its data in Amazon S3 by using the S3 Standard storage class.
A data engineer examined data access patterns to identify trends. During the first 6 months, most data files are accessed several times each day. Between 6 months and 2 years, most data files are accessed once or twice each month. After 2 years, data files are accessed only once or twice each year.
The data engineer needs to use an S3 Lifecycle policy to develop new data storage rules. The new storage solution must continue to provide high availability.
Which solution will meet these requirements in the MOST cost-effective way?

A. Transition objects to S3 One Zone-Infrequent Access (S3 One Zone-IA) after 6 month
B. Transfer objects to S3 Glacier Flexible Retrieval after 2 years.
C. Transition objects to S3 Standard-Infrequent Access (S3 Standard-IA) after 6 month
D. Transfer objects to S3 Glacier Flexible Retrieval after 2 years.
E. Transition objects to S3 Standard-Infrequent Access (S3 Standard-IA) after 6 month
F. Transfer objects to S3 Glacier Deep Archive after 2 years.
G. Transition objects to S3 One Zone-Infrequent Access (S3 One Zone-IA) after 6 month
H. Transfer objects to S3 Glacier Deep Archive after 2 years.

Answer: C

Explanation:
To achieve the most cost-effective storage solution, the data engineer needs to use an S3 Lifecycle policy that transitions objects to lower-cost storage classes based on their access patterns, and deletes them when they are no longer needed. The storage classes should also provide high availability, which means they should be resilient to the loss of data in a single Availability Zone1. Therefore, the solution must include the following steps:
? Transition objects to S3 Standard-Infrequent Access (S3 Standard-IA) after 6 months. S3 Standard-IA is designed for data that is accessed less frequently, but requires rapid access when needed. It offers the same high durability, throughput, and low latency as S3 Standard, but with a lower storage cost and a retrieval fee2.
Therefore, it is suitablefor data files that are accessed once or twice each month. S3 Standard-IA also provides high availability, as it stores data redundantly across multiple Availability Zones1.
? Transfer objects to S3 Glacier Deep Archive after 2 years. S3 Glacier Deep Archive is the lowest-cost storage class that offers secure and durable storage for data that is rarely accessed and can tolerate a 12-hour retrieval time. It is ideal for long-term archiving and digital preservation3. Therefore, it is suitable for data

files that are accessed only once or twice each year. S3 Glacier Deep Archive also provides high availability, as it stores data across at least three geographically dispersed Availability Zones1.
? Delete objects when they are no longer needed. The data engineer can specify an expiration action in the S3 Lifecycle policy to delete objects after a certain period of time. This will reduce the storage cost and comply with any data retention policies.
Option C is the only solution that includes all these steps. Therefore, option C is the correct answer.
Option A is incorrect because it transitions objects to S3 One Zone-Infrequent Access (S3 One Zone-IA) after 6 months. S3 One Zone-IA is similar to S3 Standard-IA, but it stores data in a single Availability Zone. This means it has a lower availability and durability than S3 Standard-IA, and it is not resilient to the loss of data in a single Availability Zone1. Therefore, it does not provide high availability as required.
Option B is incorrect because it transfers objects to S3 Glacier Flexible Retrieval after 2 years. S3 Glacier Flexible Retrieval is a storage class that offers secure and durable storage for data that is accessed infrequently and can tolerate a retrieval time of minutes to hours. It is more expensive than S3 Glacier Deep Archive, and it is not suitable for data that is accessed only once or twice each year3. Therefore, it is not the most cost-effective option.
Option D is incorrect because it combines the errors of option A and B. It transitions objects to S3 One Zone-IA after 6 months, which does not provide high availability, and it transfers objects to S3 Glacier Flexible Retrieval after 2 years, which is not the most cost-effective option.
References:
? 1: Amazon S3 storage classes - Amazon Simple Storage Service
? 2: Amazon S3 Standard-Infrequent Access (S3 Standard-IA) - Amazon Simple Storage Service
? 3: Amazon S3 Glacier and S3 Glacier Deep Archive - Amazon Simple Storage Service
? [4]: Expiring objects - Amazon Simple Storage Service
? [5]: Managing your storage lifecycle - Amazon Simple Storage Service
? [6]: Examples of S3 Lifecycle configuration - Amazon Simple Storage Service
? [7]: Amazon S3 Lifecycle further optimizes storage cost savings with new features
- What's New with AWS

NEW QUESTION 46
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 ofcolumns. 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 data engineer runs Amazon Athena queries on data that is in an Amazon S3 bucket. The Athena queries use AWS Glue Data Catalog as a metadata table.
The data engineer notices that the Athena query plans are experiencing a performance bottleneck. The data engineer determines that the cause of the performance bottleneck is the large number of partitions that are in the S3 bucket. The data engineer must resolve the performance bottleneck and reduce Athena query planning time.
Which solutions will meet these requirements? (Choose two.)

A. Create an AWS Glue partition inde
B. Enable partition filtering.
C. Bucketthe data based on a column thatthe data have in common in a WHERE clause of the user query
D. Use Athena partition projection based on the S3 bucket prefix.
E. Transform the data that is in the S3 bucket to Apache Parquet format.
F. Use the Amazon EMR S3DistCP utility to combine smaller objects in the S3 bucket into larger objects.

Answer: AC

Explanation:
The best solutions to resolve the performance bottleneck and reduce Athena query planning time are to create an AWS Glue partition index and enable partition filtering, and to use Athena partition projection based on the S3 bucket prefix.

AWS Glue partition indexes are a feature that allows you to speed up query processing of highly partitioned tables cataloged in AWS Glue Data Catalog. Partition indexes are available for queries in Amazon EMR, Amazon Redshift Spectrum, and AWS Glue ETL jobs. Partition indexes are sublists of partition keys defined in the table. When you create a partition index, you specify a list of partition keys that already exist on a given table. AWS Glue then creates an index for the specified keys and stores it in the Data Catalog. When you run a query that filters on the partition keys, AWS Glue uses the partition index to quickly identify the relevant partitions without scanning the entiretable metadata. This reduces the query planning time and improves the query performance1.
Athena partition projection is a feature that allows you to speed up query processing of highly partitioned tables and automate partition management. In partition projection, Athena calculates partition values and locations using the table properties that you configure directly on your table in AWS Glue. The table properties allow Athena to 'project', or determine, the necessary partition information instead of having to do a more time- consuming metadata lookup in the AWS Glue Data Catalog. Because in-memory operations are often faster than remote operations, partition projection can reduce the runtime of queries against highly partitioned tables. Partition projection also automates partition management because it removes the need to manually create partitions in Athena, AWS Glue, or your external Hive metastore2.
Option B is not the best solution, as bucketing the data based on a column that the data have in common in a WHERE clause of the user query would not reduce the query planning time. Bucketing is a technique that divides data into buckets based on a hash function applied to a column. Bucketing can improve the performance of join queries by
reducing the amount of data that needs to be shuffled between nodes. However, bucketing does not affect the partition metadata retrieval, which is the main cause of the performance bottleneck in this scenario3.
Option D is not the best solution, as transforming the data that is in the S3 bucket to Apache Parquet format would not reduce the query planning time. 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, Parquet does not affect the partition metadata retrieval, which is the main cause of the performance bottleneck in this scenario4.
Option E is not the best solution, as using the Amazon EMR S3DistCP utility to combine smaller objects in the S3 bucket into larger objects would not reduce the query planning time. S3DistCP is a tool that can copy large amounts of data between Amazon S3 buckets or from HDFS to Amazon S3. S3DistCP can also aggregate smaller files into larger files to improve the performance of sequential access. However, S3DistCP does not affect the partition metadata retrieval, which is the main cause of the performance bottleneck in this scenario5. References:
? Improve query performance using AWS Glue partition indexes
? Partition projection with Amazon Athena
? Bucketing vs Partitioning
? Columnar Storage Formats
? S3DistCp
? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide

NEW QUESTION 48
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(*J for the aggregation.
B. Change WHERE year = 2023 to WHERE extractlyear FROM sales data) = 2023.
C. Add HAVING sumfsales 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 querywill 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 53
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. Confiqure 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 55

A data engineer needs to securely transfer 5 TB of data from an on-premises data center to an Amazon S3 bucket. Approximately 5% of the data changes every day. Updates to the data need to be regularlyproliferated to the S3 bucket. The data includes files that are in multiple formats. The data engineer needs to automate the transfer process and must schedule the process to run periodically.

Which AWS service should the data engineer use to transfer the data in the MOST operationally efficient way?

A. AWS DataSync
B. AWS Glue
C. AWS Direct Connect
D. Amazon S3 Transfer Acceleration

Answer: A

Explanation:

AWS DataSync is an online data movement and discovery service that simplifies and accelerates data migrations to AWS as well as moving data to and from on-premises storage, edge locations, other cloud providers, and AWS Storage services1. AWS DataSync can copy data to and from various sources and targets, including Amazon S3, and handle files in multiple formats. AWS DataSync also supports incremental transfers, meaning it can detect and copy only the changes to the data, reducing the amount of data transferred and improving the performance. AWS DataSync can automate and schedule the transfer process using triggers, and monitor the progress and status of the transfers using CloudWatch metrics and events1.

AWS DataSync is the most operationally efficient way to transfer the data in this scenario, as it meets all the requirements and offers a serverless and scalable solution. AWS Glue, AWS Direct Connect, and Amazon S3 Transfer Acceleration are not the best options for this scenario, as they have some limitations or drawbacks compared to AWS DataSync. AWS Glue is a serverless ETL service that can extract, transform, and load data from various sources to various targets, including Amazon S32. However, AWS Glue is not designed for large-scale data transfers, as it has some quotas and limits on the number and size of files it can process3. AWS Glue also does not support incremental transfers, meaning it would have to copy the entire data set every time, which would be inefficient and costly.

AWS Direct Connect is a service that establishes a dedicated network connection between your on-premises data center and AWS, bypassing the public internet and improving the bandwidth and performance of the data transfer. However, AWS Direct Connect is not a data transfer service by itself, as it requires additional services or tools to copy the data, such as AWS DataSync, AWS Storage Gateway, or AWS CLI. AWS Direct Connect also has some hardware and location requirements, and charges you for the port hours and data transfer out of AWS.

Amazon S3 Transfer Acceleration is a feature that enables faster data transfers to Amazon S3 over long distances, using the AWS edge locations and optimized network paths. However, Amazon S3 Transfer Acceleration is not a data transfer service by itself, as it requires additional services or tools to copy the data, such as AWS CLI, AWS SDK, or third-party software. Amazon S3 Transfer Acceleration also charges you for the data transferred over the accelerated endpoints, and does not guarantee a performance improvement for every transfer, as it depends on various factors such as the network conditions, the distance, and the object size. References:
? AWS DataSync
? AWS Glue
? AWS Glue quotas and limits
? [AWS Direct Connect]
? [Data transfer options for AWS Direct Connect]
? [Amazon S3 Transfer Acceleration]
? [Using Amazon S3 Transfer Acceleration]

NEW QUESTION 57

A data engineer needs to use AWS Step Functions to design an orchestration workflow. The workflow must parallel process a large collection of data files and apply a specific transformation to each file.

Which Step Functions state should the data engineer use to meet these requirements?

A. Parallel state
B. Choice state
C. Map state
D. Wait state

Answer: C

Explanation:

Option C is the correct answer because the Map state is designed to process a collection of data in parallel by applying the same transformation to each element. The Map state can invoke a nested workflow for each element, which can be another state machine ora Lambda function. The Map state will wait until all the parallel executions are completed before moving to the next state.

Option A is incorrect because the Parallel state is used to execute multiple branches of logic concurrently, not to process a collection of data. The Parallel state can have different branches with different logic and states, whereas the Map state has only one branch that is applied to each element of the collection.

Option B is incorrect because the Choice state is used to make decisions based on a comparison of a value to a set of rules. The Choice state does not process any data or invoke any nested workflows.

Option D is incorrect because the Wait state is used to delay the state machine from continuing for a specified time. The Wait state does not process any data or invoke any nested workflows.
References:
? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 5: Data Orchestration, Section 5.3: AWS Step Functions, Pages 131-132
? Building Batch Data Analytics Solutions on AWS, Module 5: Data Orchestration, Lesson 5.2: AWS Step Functions, Pages 9-10
? AWS Documentation Overview, AWS Step Functions Developer Guide, Step Functions Concepts, State Types, Map State, Pages 1-3

NEW QUESTION 62
A data engineer needs Amazon Athena queries to finish faster. The data engineer notices that all the files the Athena queries use are currently stored in uncompressed .csv format. The data engineer also notices that users perform most queries by selecting a specific column.
Which solution will MOST speed up the Athena query performance?

A. Change the data format from .csvto JSON forma
B. Apply Snappy compression.
C. Compress the .csv files by using Snappy compression.
D. Change the data format from .csvto Apache Parque
E. Apply Snappy compression.
F. Compress the .csv files by using gzjg compression.

Answer: C

Explanation:
Amazon Athena is a serverless interactive query service that allows you to analyze data in Amazon S3 using standard SQL. Athena supports 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, changing the data format from CSV to Parquet and applying Snappy compression will most speed up the Athena query performance. Parquet is a column- oriented format that allows Athena to scan only the relevant columns and skip the rest, reducing the amount of data read from S3. Snappy is a compression algorithm that reduces the data size without compromising the query speed, as it is splittable and does not require decompression before reading. This solution will also reduce the cost of Athena queries, as Athena charges based on the amount of data scanned from S3.
The other options are not as effective as changing the data format to Parquet and applying Snappy compression. Changing the data format from CSV to JSON and applying Snappy compression will not improve the query performance significantly, as JSON is also a row- oriented format that does not support columnar access or encoding techniques. Compressing the CSV files by using Snappy compression will reduce the data size, but it will not improve the query performance significantly, as CSV is still a row-oriented format that does not support columnar access or encoding techniques. Compressing the CSV files by using gzjg compression will reduce the data size, but it willdegrade the query performance, as gzjg is not a splittable compression algorithm and requires decompression before reading. References:
? Amazon Athena
? Choosing the Right Data Format
? AWS Certified Data Engineer - Associate DEA-C01 Complete Study Guide, Chapter 5: Data Analysis and Visualization, Section 5.1: Amazon Athena

NEW QUESTION 64
......